Advanced Linear Models

Lecture Notes

Aaron R. Caldwell

Table of contents

Preface

This is a Quarto book of the course notes for Biostatistics II: Advanced Linear Models. The book can be viewed online as HTML or downloaded as pdf.

1 Review of Introductory Inference

This section is intended to review the basic knowledge you'll need for this course.

1.1 Review: Inference

There is some expectation of knowledge that you should know from your previous statistics course.

Key concepts you are expected to have knowledge of:

- Populations and samples
- Probability
- Random variables
- Probability distributions
  - probability density functions
  - cumulative distribution functions
  - Normal distribution
  - t distribution
  - F distribution
- Sampling distributions
- Confidence intervals and hypothesis tests
  - one-sample mean
  - two-sample means

There will be a brief review of some of the more topics and a set of materials will be linked to at the end of this appendix.

1.2 General Idea of Inference

We will cover some of the topics you should know from a previous introductory course in statistics. There is a general procedure that we can conceptualize.

- There is a general target population we are interested in in some way.

- We have certain questions we want to ask about this population.
- We figure out what can we quantify from the population and how those quantfications may answer our questions.
- We collect our data/measurements from the population.
    - Sometimes (maybe often) the way we collect data changes the exact nature of the population we are interested in.
- We perform statistical analyses/inference on the population to see how the data answers our questions.
- We communicate our findings in some way… Typically.

1.2.1 Populations and Parameters: Means and Standard Deviations

If I have some population Y:

$$\bar{Y} \rightarrow RandomVariable \rightarrow Population$$

- μ: mean
- σ: standard deviation

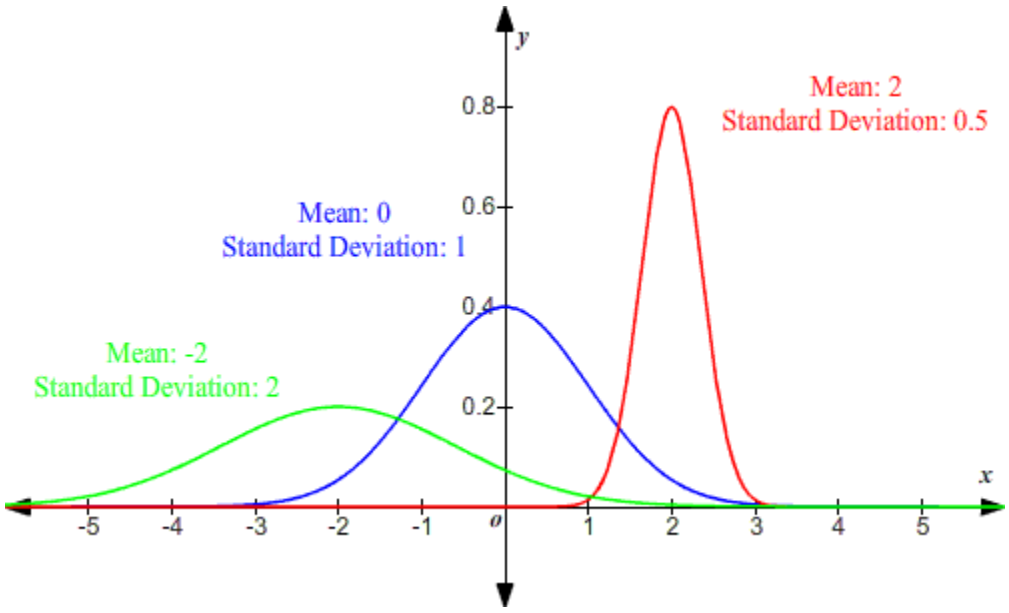


Figure 1.1: Different means and standard deviations

1.2.2 Estimating The Mean and Standard Deviation

We estimate the mean of a population by taking a sample. We assume simple random samples; you might want to look up what the means if you forgot.

The estimate of a population mean *mu* is the *sample mean* which is typically denoted by $\bar{y}$.

$$\bar{y} = \frac{\Sigma y_i}{n} \rightarrow estimate\mu$$

The estimate of the population standard deviation σ is the *sample standard deviation*, denoted by s.

$$\hat{\sigma} = s = \sqrt{\frac{\Sigma(y_i - \bar{y})^2}{n-1}}$$

We refer to these statistics as **point estimates**. This term is used to emphasize the fact that we have some fixed number when we do the calculation.

1.3 Central Limit Theorem, Standard Errors, and Uncertainty

1.3.1 Standard Error

The standard measure of reliability for $\bar{y}$ is the **standard error**.

$$\sigma_{\bar{y}} = \frac{\sigma}{\sqrt{n}}$$

1.3.1.1 Estimated SE

$$SE_{\bar{y}} = \frac{s}{\sqrt{n}}$$

This is our measurement of relative uncertainty of the sample mean.

1.3.2 Central Limit Theorem

The standard error is an important part of the **Central Limit Theorem (CLT)**. The bare-bones of the CLT is:

$$\bar{y} \sim N(\mu, \frac{\sigma}{\sqrt{n}})$$

- SE: $\frac{s}{\sqrt{n}}$
- t-distribution

The CLT allows us to, under certain assumptions, figure a range of likely values for the true mean μ. These assumptions are either:

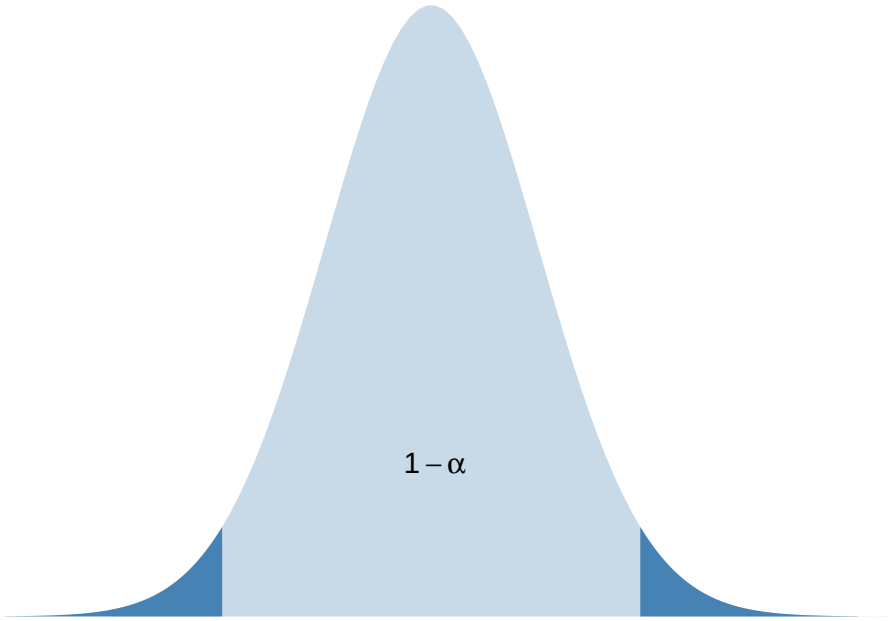
1. The population we take the sample from is approximately normally distributed, or
2. We have a sufficiently large sample size that we can ignore assumption 1. "Sufficiently large" is typically characterized as $n > 30$, but that rule-of-thumb would depend on the population distribution.

A link to a demonstration is here: https://gallery.shinyapps.io/CLT_mean/

1.4 Confidence Intervals for the Mean

This is the $(1 - \alpha)100\%$ confidence interval for μ.

$$\bar{y} \pm t_{\alpha/2,df} \frac{s}{\sqrt{n}}$$



- Margin of Error
- The confidence level $(1-\alpha)\cdot 100\%$ is the reliability of a computed interval.
- $t_{\alpha/2,df}$ is the value from the t distribution with a right tail area of $\alpha/2$ and degrees of freedom $df = n-1$. The degrees of freedom formula will change depending on what is being estimated.

1.4.1 General Form for Confidence Intervals

There are many situations where we are estimating some parameter θ of a population.

Let $\hat{\theta}$ represent the estimate of θ.

$$\hat{\theta} \pm t_{\alpha/2,df} \cdot SE_{\hat{\theta}}$$

1.5 Hypothesies Tests

A hypothesis test is a statistics procedure that is meant to assess the validity that a population parameter θ differs from some predefined value θ_0.

The basic procedure is this:

1. Hypotheses about θ

- $H_0 : \theta = \theta_0$
- $H_1 : \theta \neq \theta_0$

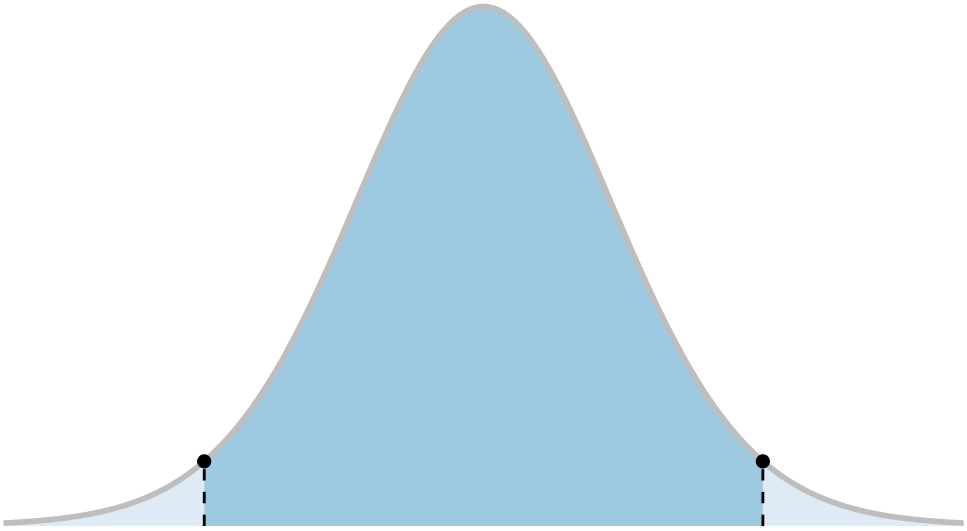
2. Collect data and estimate θ with $\hat{\theta}$

3. Test statistic

$$t_s = \frac{\hat{\theta} - \theta_0}{SE_{\hat{\theta}}}$$

4. p-value $p = Pr(T_y \geq |t_s|)$

- $p < \alpha$, if yes then reject H_0
- Remember, α is the desired type 1 error rate



t distribution with 18 degrees of freedom

1.6 Review Videos (courtesy of JB Statistics and Crash Course)

Given that this is material that you are expected to know for this class, I am only putting this here as a reminder, and for you to gauge yourself on how much you remember.

Should you feel that you need a refresher, the website and book of Balka (n.d.) provides a fairly thorough break of any and all the topics you should know about coming into this course from Biostatistics 101.

I've provided links to some of the videos relevant to the pre-requisite material.

1.6.1 Probability Distributions

- Crash Course Distributions
- Normal Distribution:
  - An Introduction to the Normal Distribution
  - Crash Course on the Normal Distribution
  - Standardizing Normally Distributed Random Variables
- t-distribution:
  - An Introduction to the Chi-Square Distribution
  - An Introduction to the t Distribution (Includes some mathematical details)
  - Intro to the t Distribution (non-technical)
- F-distribution: An Introduction to the F Distribution

1.6.2 Sampling Distributions and the Central Limit Theorem (CLT)

- Sampling Distributions: Introduction to the Concept
- The Sampling Distribution of the Sample Mean
- Introduction to the Central Limit Theorem
- Central Limit Demonstration App
- Crash Course Z-score

1.6.3 Confidence Intervals

- Crash Course on Confidence Intervals
- Introduction to Confidence Intervals
- Deriving a Confidence Interval for the Mean
- Confidence Intervals for One Mean: Sigma Not Known (t Method)
- Intro to the t Distribution (non-technical)

1.6.4 Hypothesis Tests

- Crash Course p-values: part 1
- Crash Course p-values: part 2
- Crash Course p-values: part 3
- An Introduction to Hypothesis Testing
- t Tests for One Mean: Introduction
- t Tests for One Mean: An Example

2 Data and Models

```
knitr::opts_chunk$set(echo = FALSE, tidy = TRUE,
                      cache = FALSE,
                      message = FALSE, WARNING = FALSE)
# Very standard packages
library(graphics)
library(ggplot2)
library(tidyverse)
library(knitr)
library(readr)
library(MASS)
library(plotly)
library(flextable)
# I do not want default theme.
old.theme <- theme_get()
theme_set(theme_bw())
```

2.1 Data

2.1.1 Variables and Observations

Let's talk about what we mean by data, in this course.

Data is composed of **variables** and **observations**.

Example: We have a patient. We measure their blood pressure. It is *observed* to be 133/86.

Variable: Blood pressure

Observation: 133/86

Or we could reformulate this

Variables: Systolic blood pressure and diastolic blood pressure.

Observation(s): 133 and 86

2.1.2 Heart data introduction

```
heart <- read_csv(here::here("datasets", "Heart.csv"))

as_flextable(head(heart))
```

age	sex	chestPain	restSysBP	cholesterol	fastBldSgr	restECG
numeric	character	character	numeric	numeric	numeric	numeric
63	Male	typical	145	233	1	2
67	Male	asymptomatic	160	286	0	2
67	Male	asymptomatic	120	229	0	2
37	Male	nonanginal	130	250	0	0
41	Female	nontypical	130	204	0	2
56	Male	nontypical	120	236	0	0

n: 6

2.1.3 Heart Disease Data Dictionary

A data dictionary explains what the "names" of variables in a dataset mean. For the heart data:

- `age`: The patient's age in years
- `sex`: The patient's sex, `Male` or `Female`.
- `chestPain`: The chest pain experienced
  - `typical`: typical angina
  - `nontypical`': abnormal angina
  - `nonaginal`: non-anginal pain
  - `asymptomatic`: no pain
- `restSysBP`: systolic blood pressure upon admission to hospital in mm Hg
- `cholesterol`: The patient's cholesterol measurement in mg/dl
- `fastBldSgr`: indicator for whether the paitient's fasting blood sugar was greater than 120 mg/dl: `1` if yes, `0` if no.
- `restECG`: Resting electrocardiographic measurement
  - `0`: normal
  - `1`: having ST-T wave abnormality
  - `2` showing probable or definite left ventricular hypertrophy by Estes' criteria
- `maxHR`: The patient's maximum heart rate achieved during controlled exercise
- `exAng`: Exercise induced angina: `1` if yes, `0` if no
- `slope`: the slope of the peak exercise ST segment
  - `1` if slope is positive
  - `2` if slope is approximately 0
  - `3` if slope is negative
- `majorVessels`: The number of major vessels (0-3) colored in fluoroscopy.
- `disease`: Indicates whether a patient had heart disease: `Yes` if yes, `no` if no.

2.2 Mathematical Models

2.2.1 input and output

The whole

The general form of a model is deceptively simple:

$$input \to output.$$

We have some information, the *input*, we use some process, $\to$, in order to get some information, the *output*.

Model: *put a quarter in the gumball machine, turn the knob, and a gumball comes out.*

Figure 2.1: We have quarter as input to the process, gumball machine, which allows to yield a

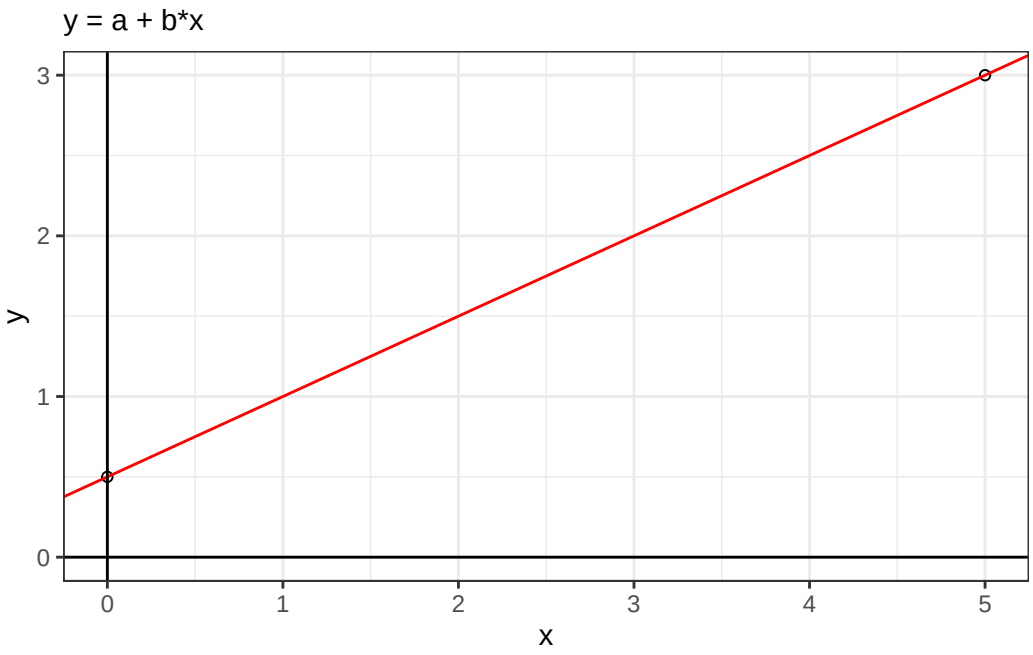
2.2.2 Mathematical models

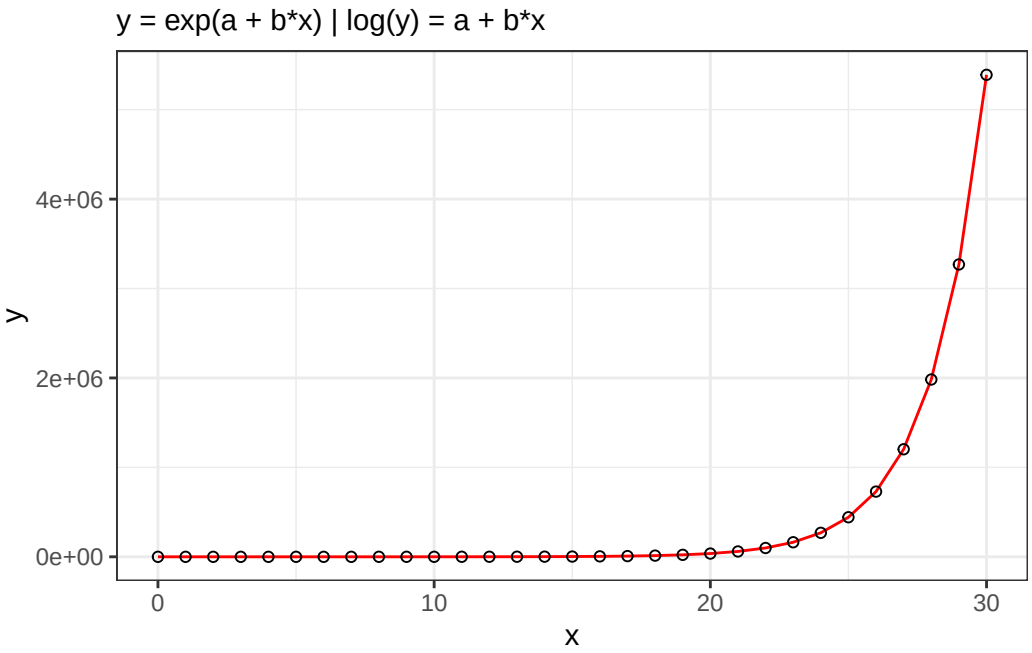
We are doing math here. We will represent our input as x, and our output as y. We put our input x into some function $g()$.

$$y = g(x)$$

- x is the input
- g is the $\rightarrow$
- y is the output

$$y = a + b \cdot x$$

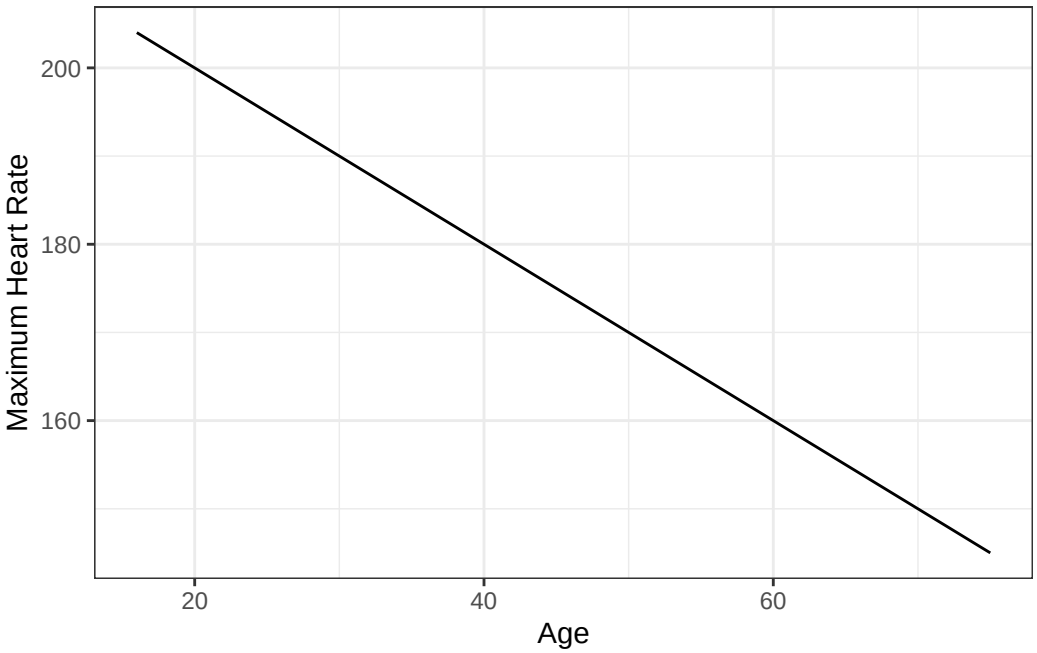




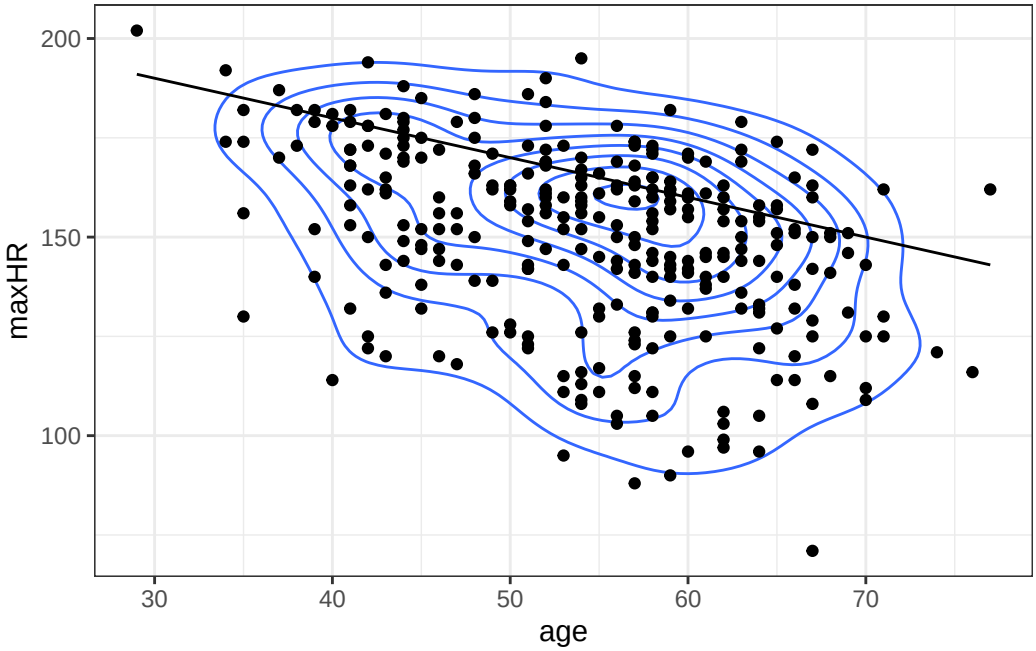
2.2.3 Heart Model

The Mayo Clinic says "You can calculate your maximum heart rate by subtracting your age from 220".

- $maxHR = 220 - age$
- $y = 220 - x$
  - y is maxHR
  - x is age


Maximum Heart Rate
200
180
160
20
40
60
Age

2.2.3.1 What about with the real data?



2.3 Statistical models and Error

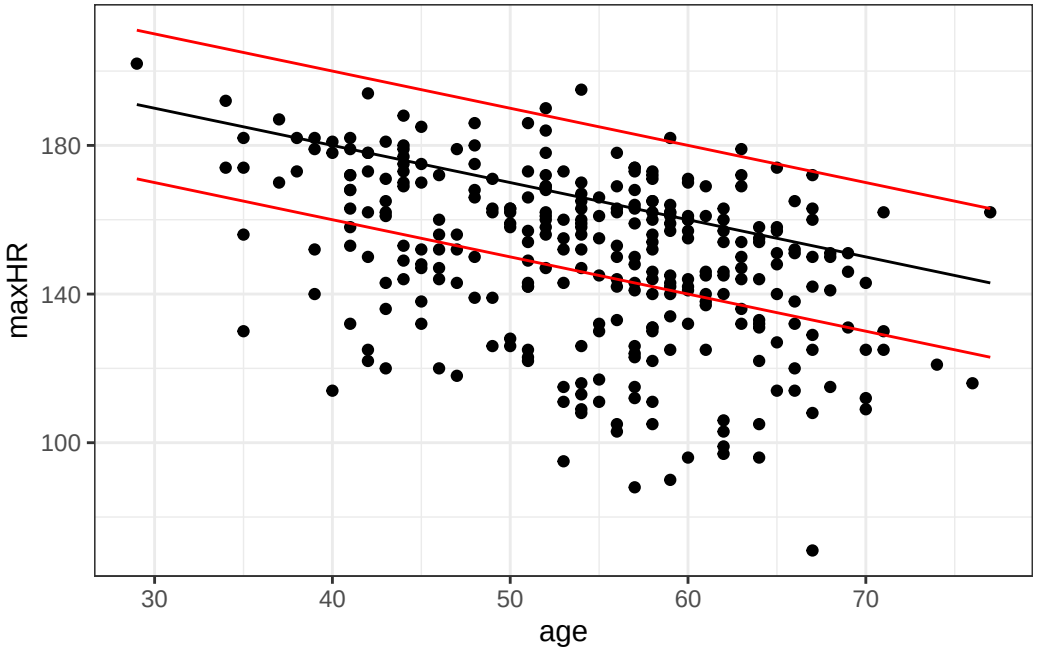
$$y = g(x) + \epsilon$$

- y = response
- x = predictor
- g(x) = function
- ϵ = error or variability in the model

2.3.1 Heart example

The Mayo Clinic also specifies "You may have a higher or lower maximum heart rate, sometimes by as much as 15 to 20 beats per minute".

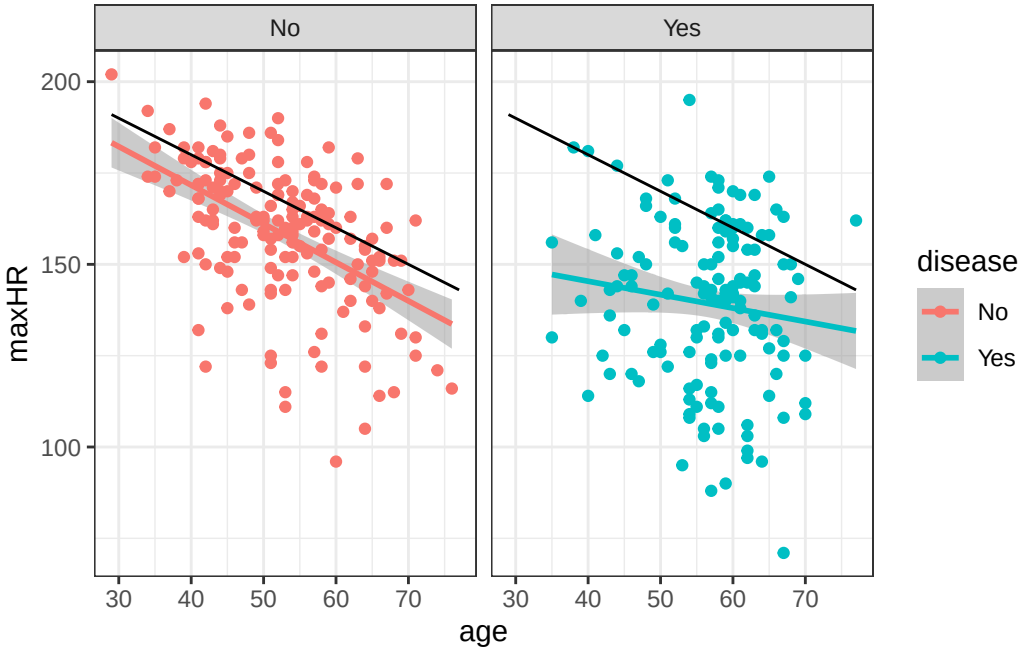
$$MaxHR = 220 - age \pm 20$$



What's going on here?

1. The guidelines from the Mayo Clinic apply mainly to the overall population of adults (age 16+).
2. This data is based on a study about *heart disease*.
3. Primarily, there are two groups in the data: those with heart disease, and those without.
4. Maybe heart disease has an effect.

2.3.1.1 Grouping Means



2.3.2 Conditional Means vs Unconditional Means

Let's concentrate on the formula for basic statistical models.

$$y = g(x) + \epsilon$$

2.3.2.1 Simplest Example

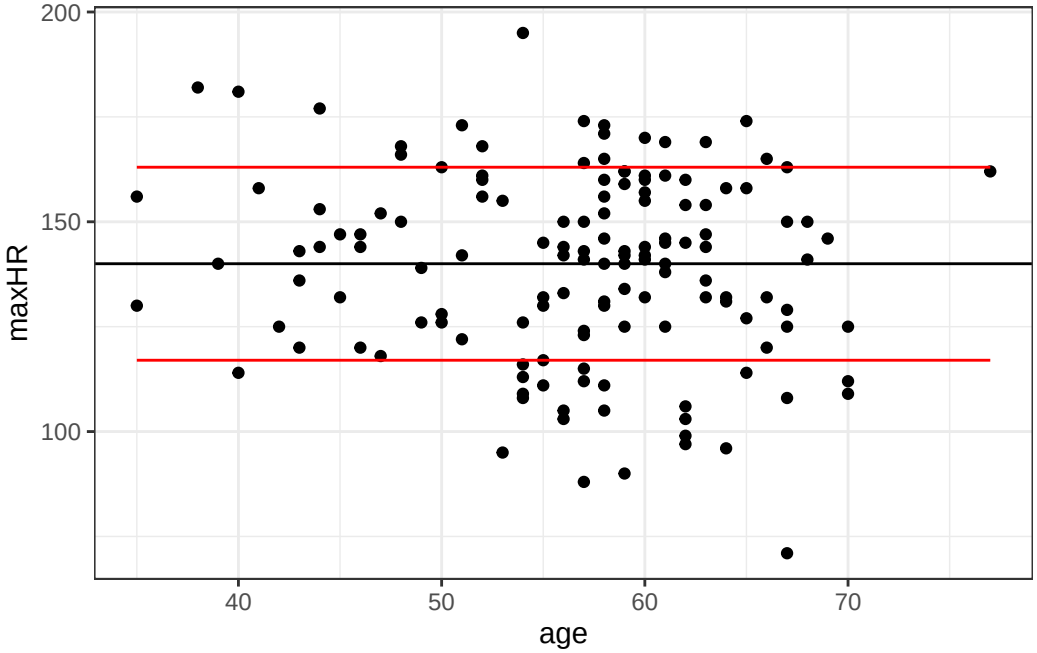
Unconditional Mean

$$y = \mu + \epsilon$$

2.3.2.2 Simple Model With Disease

$$y = 140 \pm 23$$

Why might we use this model?



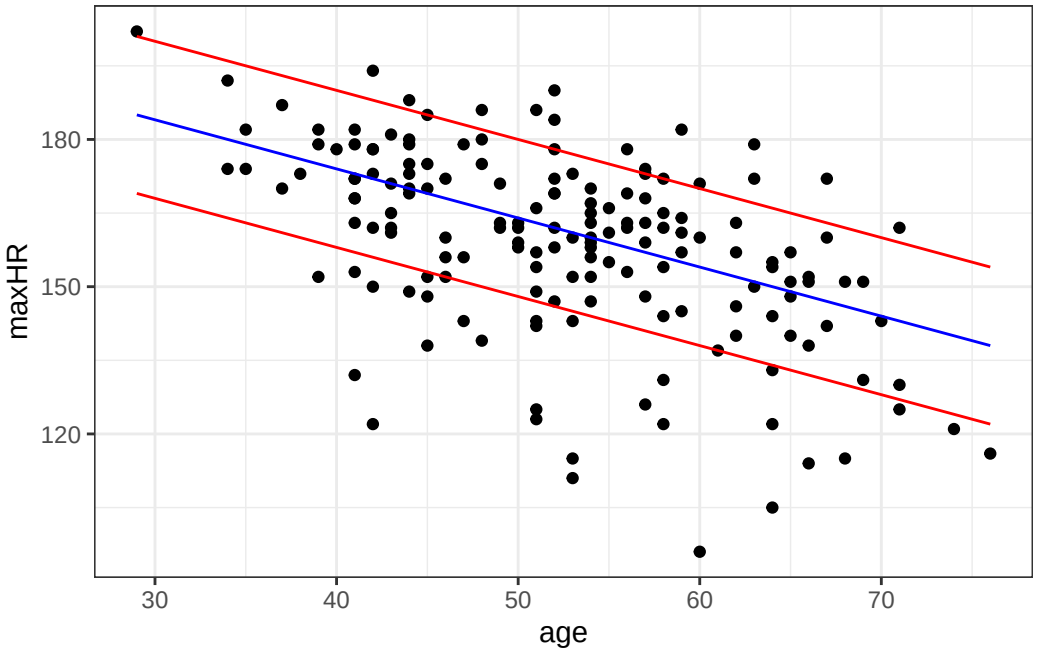
Percent 140 +/- 23
82.01439

2.3.2.3 Model for Those Without Disease

The average maxHR should be about 214 - age, the standard deviation of that model is about 16

We denote this as:

$$y = 214 - x \pm 16$$



Percent 214 - x +/- 16
0.7256098

2.4 Linear models

2.4.1 Simple linear models: one predictor variable.

The simple linear model says the $\mu_{y|x}$ is a line that depends on x based on a y-intercept which we denote by β_0 and a slope which we denote β_1.

$$\mu_{y|x} = \beta_0 + \beta_1 x$$

- Conditional mean (deterministic)
- β_0: y-intercept
- β_1: slope

$$y = \beta_0 + \beta_1 x + \epsilon = \mu_{y|x} + \epsilon$$

- ϵ is the error in the model (noted as "randomness" in some lecture videos[1])

2.4.2 Linear models with more than one predictor variable

A linear model in general means a model that can be written as a sum of variables and coefficients:

$$y = \beta_0 + \sum_{i=1}^{k} \beta_i x_i + \epsilon$$

- $\mu_{y|x}$ deterministic
- ϵ is the model error

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + ... + \beta_k x_k + \epsilon$$

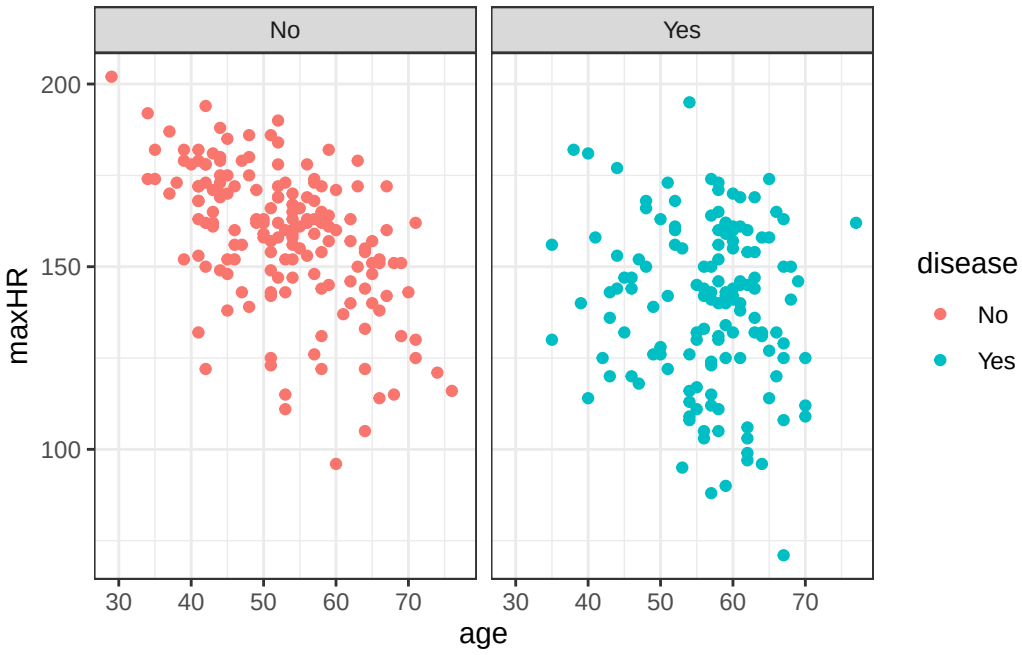
[1] We should

3 Measuring Association

```
knitr::opts_chunk$set(echo = FALSE, tidy = TRUE, cache = FALSE, message = FALSE, WARNING = FA
# Very standard packages
library(graphics)
library(ggplot2)
library(tidyverse)
library(knitr)
library(MASS)

# Not so standard
library(gridExtra) # for grid.arrange(), grids of plots in ggplot without facet_grid
library(ggpubr) #for stat_cor function which adds correlation coefficient to ggplot
library(energy) # distance correlation function and t-test in here
library(scatterplot3d) # for easy/boring scatterplots that work in PDF knit

# Good for running
library(ggstatsplot)
# Globally changing the default ggplot theme.

## store default
old.theme <- theme_get()

## Change it to theme_bw(); i don't like the grey background. Look up other themes to find y
theme_set(theme_bw())
```

3.1 Getting Started

Let's look back at that heart data.



3.2 Linear Correlation

We refer to the strength of relation between two variables to be their **correlation**. There are a few common ways to measure correlation. The most common is the following.

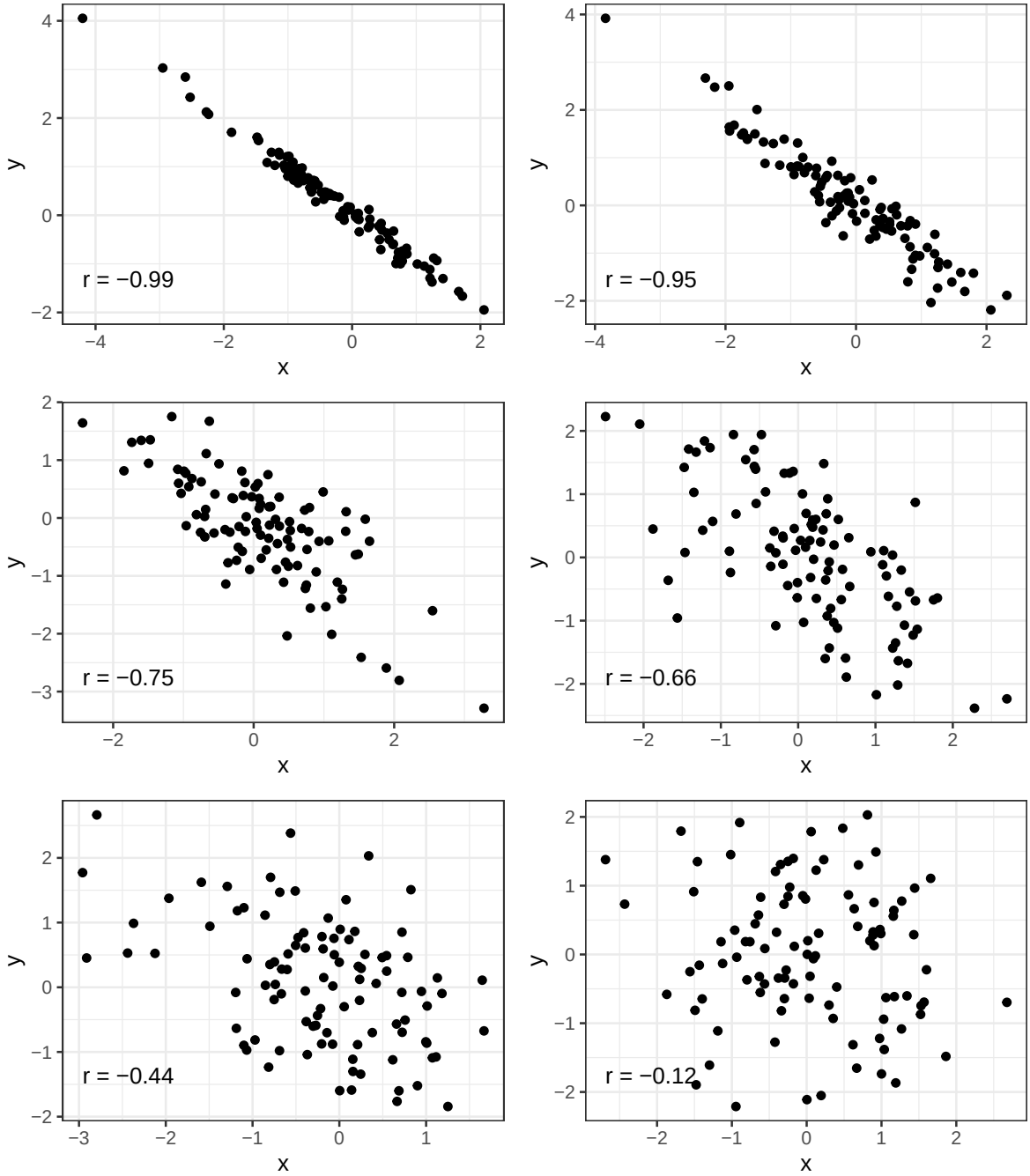
Pearson product-moment correlation:

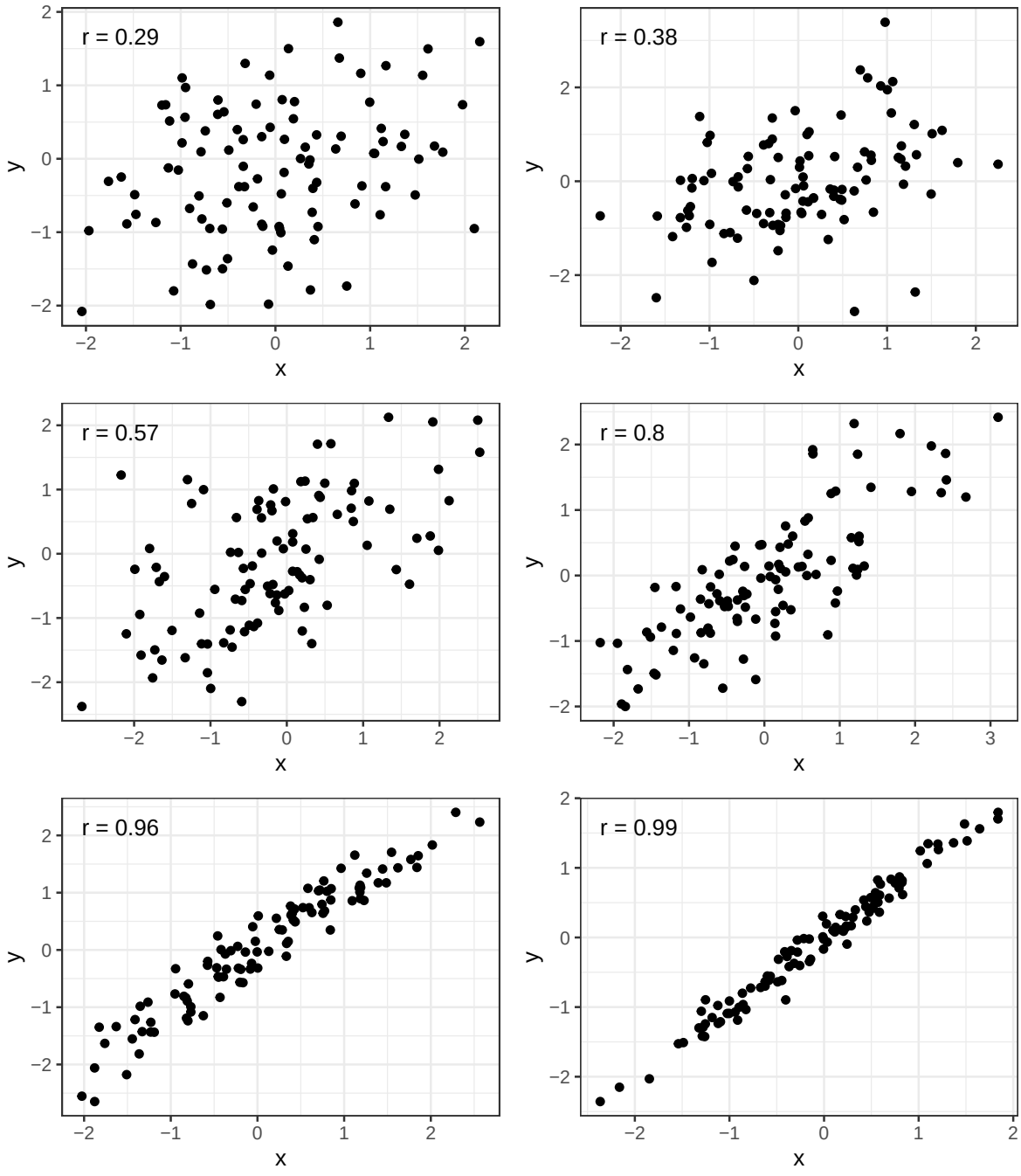
$$r = \frac{\sum_{i=1}^{n} \left(X_i - \bar{X}\right)\left(Y_i - \bar{Y}\right)}{\sqrt{\sum_{i=1}^{n} \left(X_i - \bar{X}\right)^2 \sum_{i=1}^{n} \left(Y_i - \bar{Y}\right)^2}}$$

Here are some of the common properties of r:

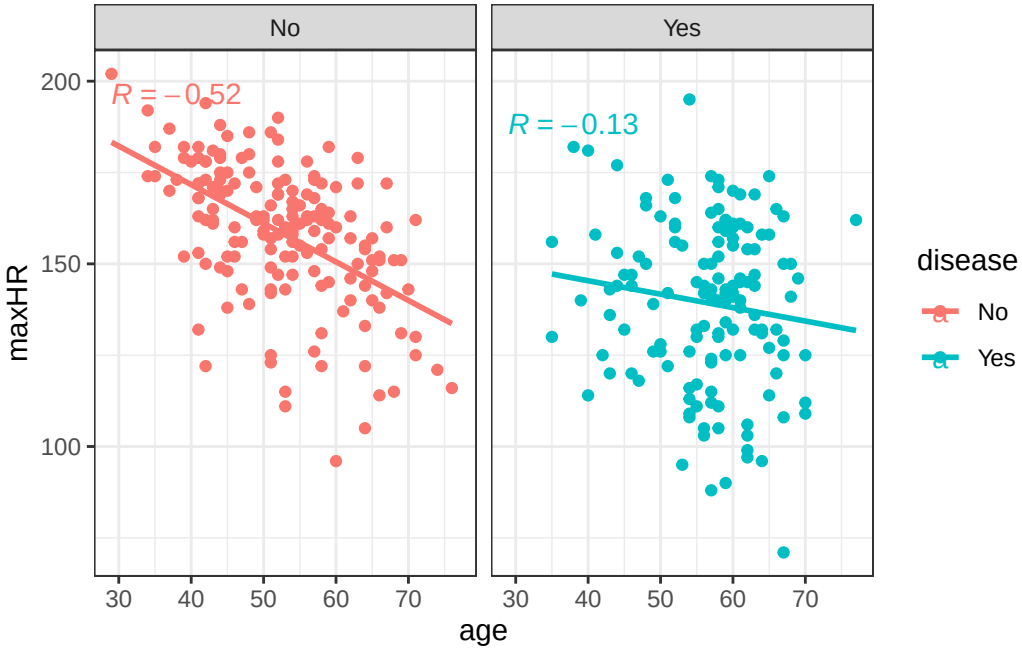
- It can take on a value from -1 to 1.
- If it is negative, then there is a "negative" relation between x and y which means as x increases, y decreases.
- If it is positive, then there is a "positive" relation between x and y which means as x increases, y increases.
- The closer to -1 or 1, the closer the x and y observations follow a straight **line**.
- The above calculation is an estimate of what is the true correlation between two random variables/populations X and Y.
- This true correlation is denoted by ρ
- Thus, r is a *point estimate* (remember that term?) of ρ (i.e., it is $\widehat{\rho}$).

3.2.1 Correlation Strength Examples




r = 0.29
r = 0.38
r = 0.57
r = 0.8
r = 0.96
r = 0.99
x
y

3.2.2 Linear correlation of the heart data



3.2.3 Correlation does not imply causation

Messerli, F. H. (2012). Chocolate Consumption, Cognitive Function, and Nobel Laureates. New England Journal of Medicine, 367(16), 1562-1564. doi:10.1056/nejmon1211064

The author tried to assert that this point towards the idea that chocolate increases cognitive function.

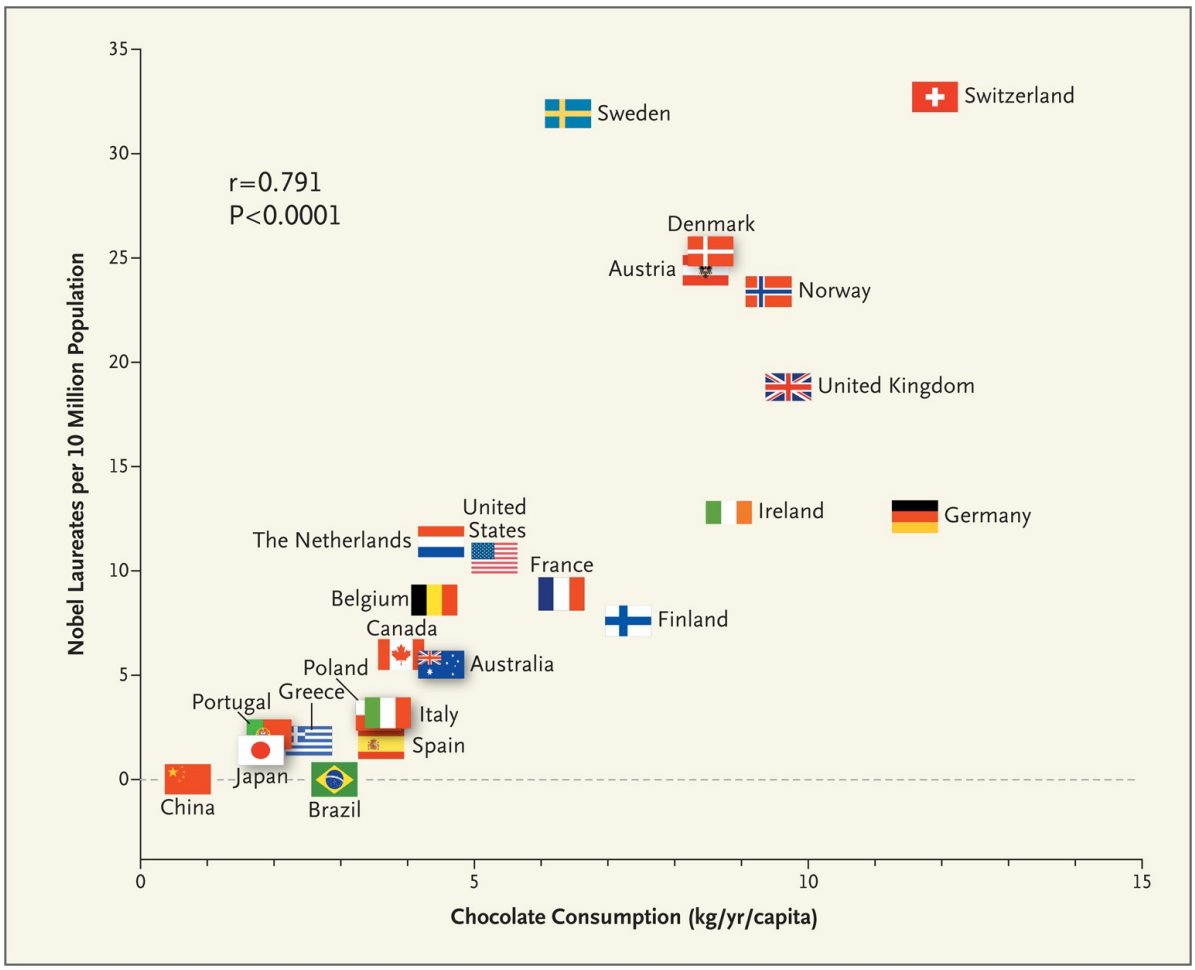
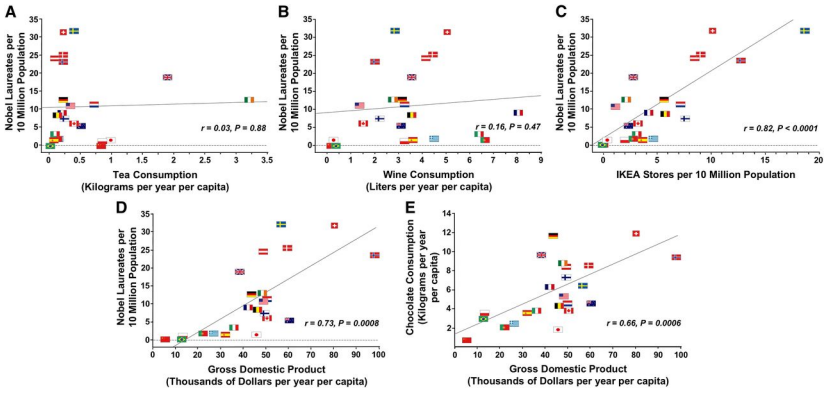


Figure 3.1: Choclate consumption and Nobel Prizes

A rebuttal from various authors produced the following graphs. What might be in common?



3.2.4 cOrReLIAtIoN dOeS nOt ImPlY cAuSaTiOn

I feel like this is a fall back phrase for those that just want some sort of easy yes/no kind of answer. "It's a correlation? Then this result is worthless." This is incredibly lazy logic.

- Do not dismiss correlations out of hand.
- Use them to ask questions!
- Correlation may not imply causation but it does imply a connection.
- Finding the connection the cause of the non-cause would be quite interesting in a lot of scenarios.

Correlation $\neq$ causation can be abused.

An article from the Science Based Medicine blog says:

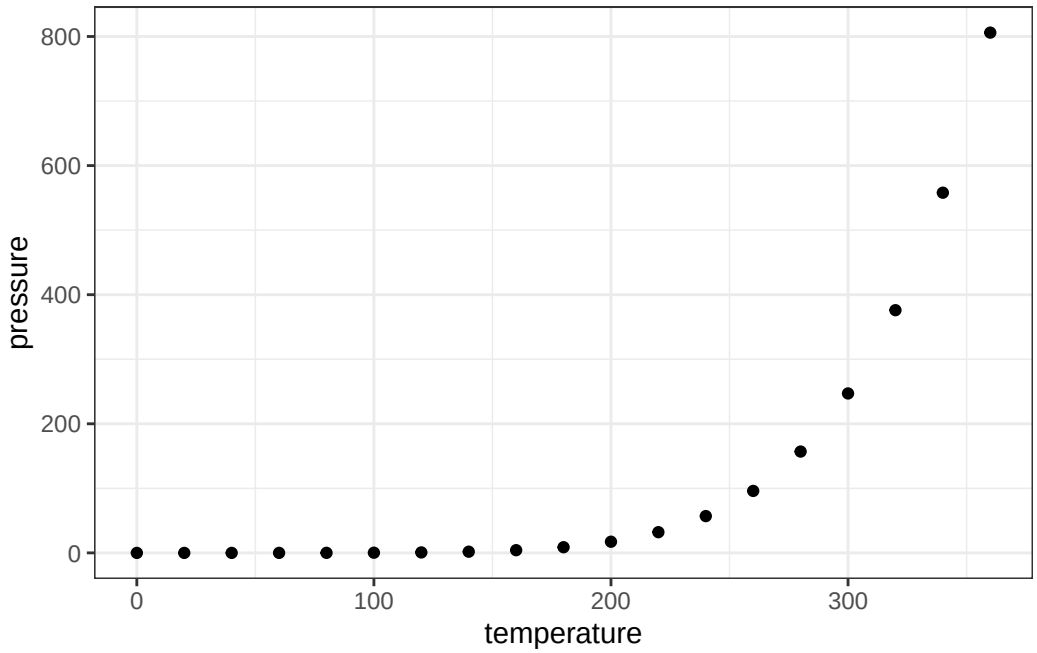
> "For example, the tobacco industry abused this fallacy to argue that simply because smoking correlates with lung cancer that does not mean that smoking causes lung cancer. The simple correlation is not enough to arrive at a conclusion of causation, but multiple correlations all triangulating on the conclusion that smoking causes lung cancer, combined with biological plausibility, does."

It should be noted that other methods can, and should, be used to derive causation

3.3 Non-linear correlation

First, let's look at a purely deterministic system.

- `pressure` is vapor pressure of mercury in mm Hg. (pressure inside a closed system)
- `temperature` is the temperature in $^{\circ}C$.



3.3.1 True Pressure Equation

How precise? Well here is the equation for calculating the vapor pressure of an element or molecule.

$$P = 10^{\left(A - \frac{B}{C+T}\right)}$$

- P is vapor pressure.
- A, B, and C are constants based on the temperature scale, pressure scale, and element/molecule.
- T is the temperature.

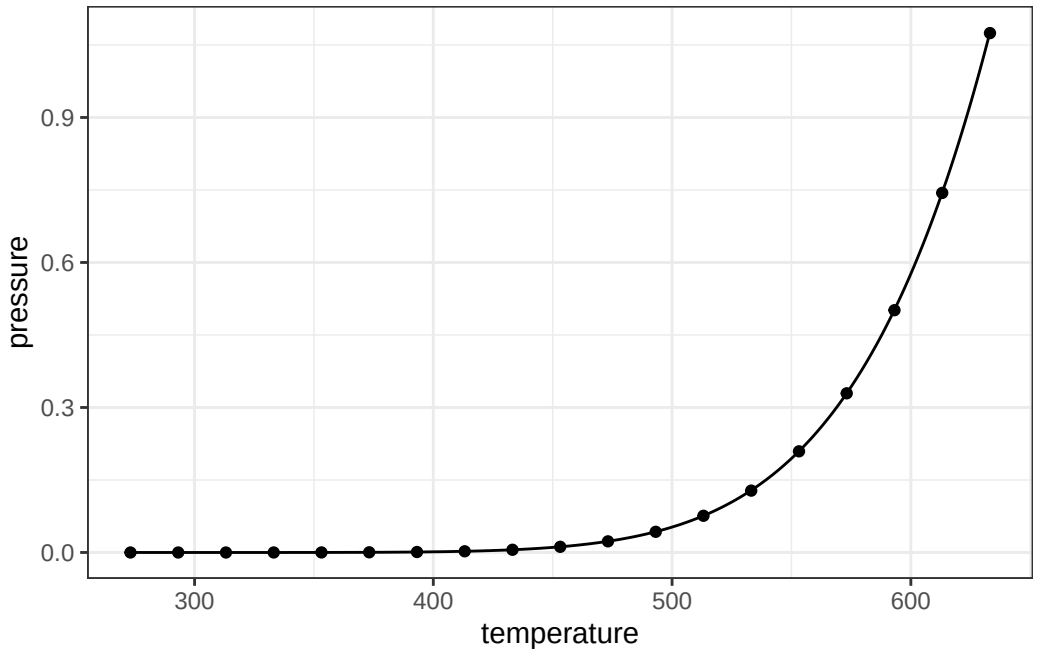
The NIST reports the constants for mercury when pressured is measured in *bar* and temperature is measured in Kelvin (K)

- $A = 4.85767$
- $B = 3007.129$
- $C = -10.001$

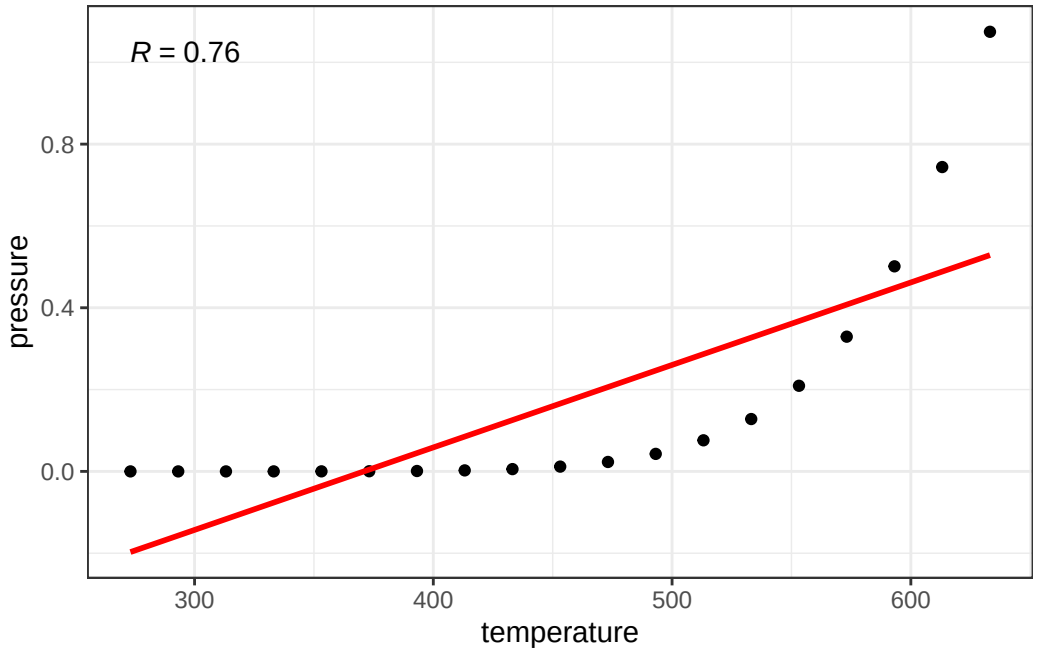
Notice these are constants. No randomness here, no error here.

3.3.2 Using the Equation

Next, let's plot the equation to our points.

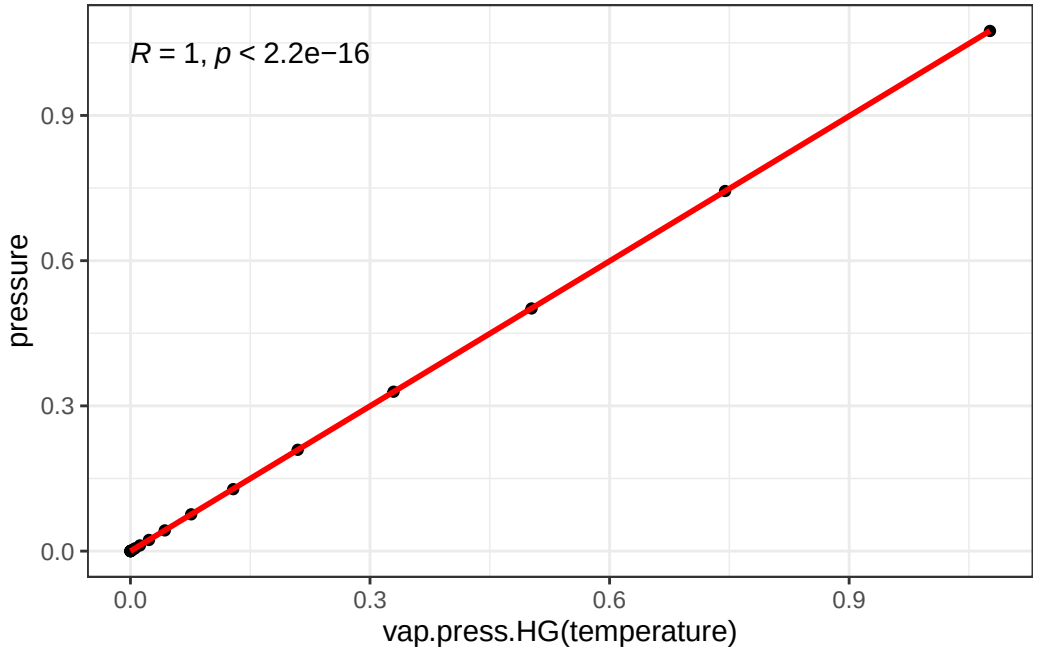


The relationship between vapor pressure and temperature seems to be perfectly accounted for by this equation. If we were to have a way to measure the strength of the relationshipo, it would hopefully reflect that perfect relation.



3.3.3 Using Transformations

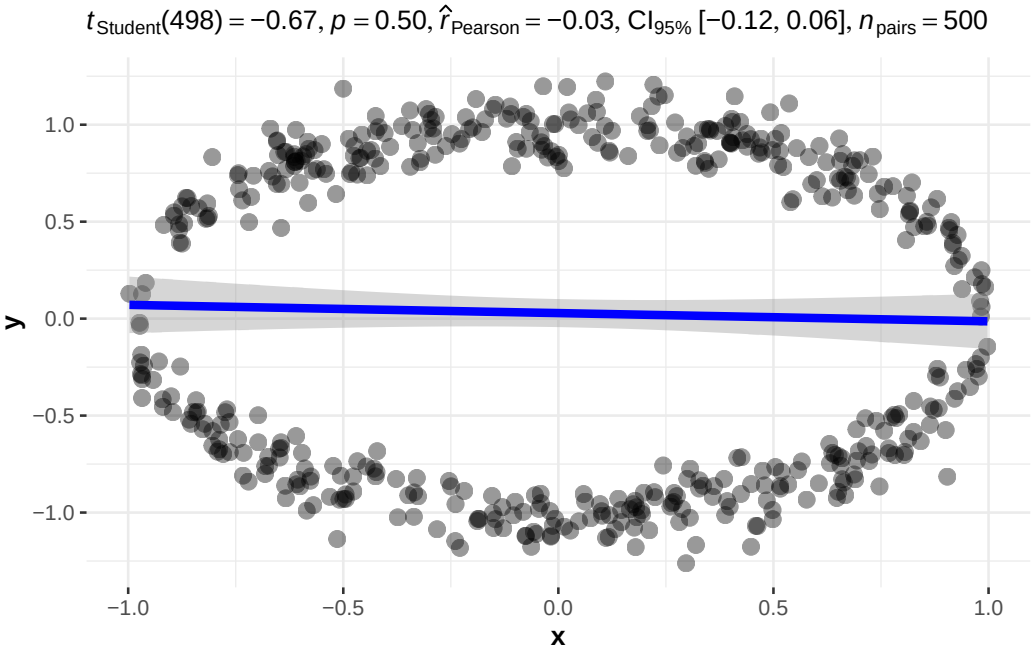
$$P = 10^{\left(4.85767 - \frac{3007.129}{-10.001 + T}\right)}$$



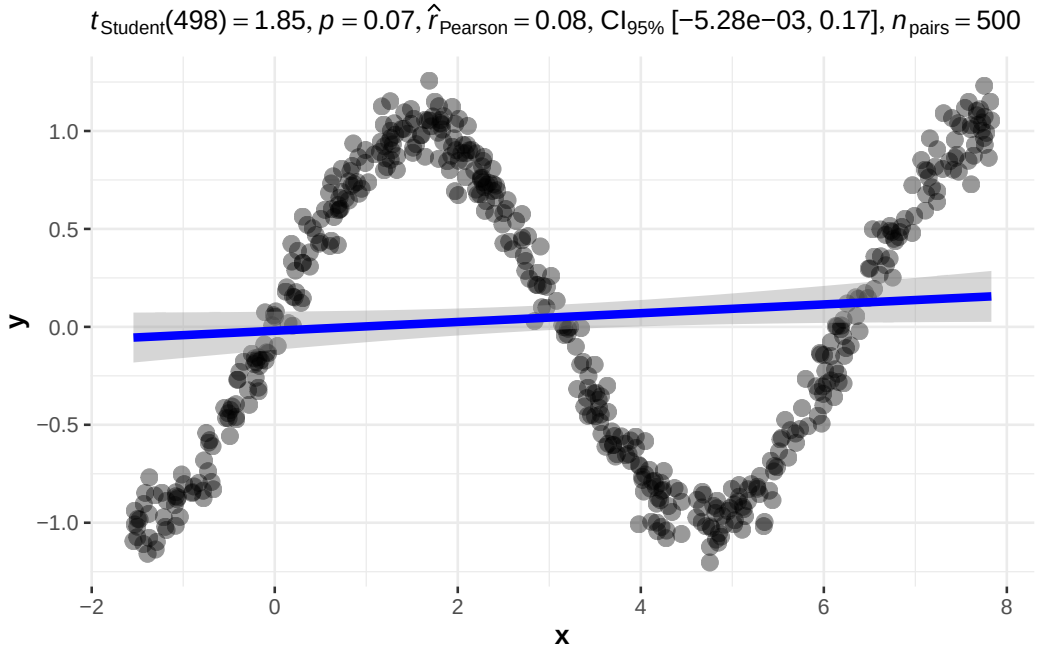
3.4 Zero Linear Relation Examples

1. Data falling on a circle. This is a "non-functional" relatioship. The mathematical definition of a function stipulates only one value of $f(x)$ is the outcome for a value of x. Another way of saying this is that one input value x should result in one and only one output value y. (Remember that horizontal line rule?) On a circle, two values are possible for an input value of x, except the leftmost and rightmost points of the circle.
2. Data from a sine wave.
3. Data from a quadratic function.

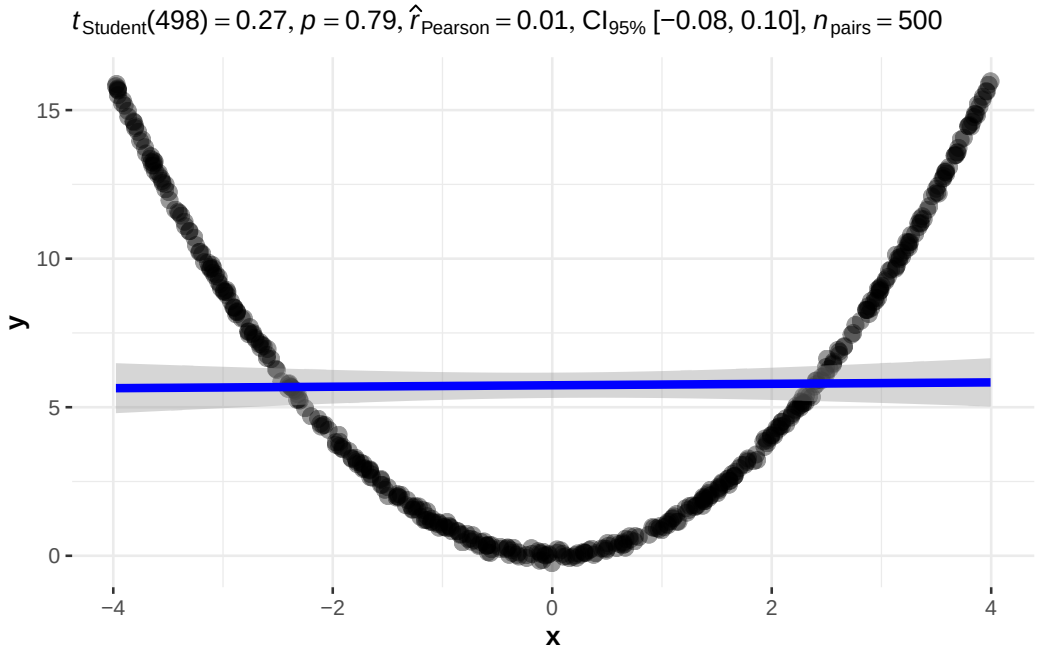
3.4.1 Circle



3.4.2 Sine Wave



3.4.3 Quadratic



3.5 Kendall's τ: A Correlation that identifies certain non-linear

Concordance: $(x_i - x_j)(y_i - y_j)$ is positive. The pair of points indicate a positive trend.

Discordance:$(x_i - x_j)(y_i - y_j)$ is negative. The pair of points indicate a negative trend.

$$\hat{\tau} = \frac{2}{n(n-1)} \sum_{i<j} sgn\left(x_i - x_j\right) sgn\left(y_i - y_j\right)$$

$$sgn(x) = \begin{cases} 1, & x > 0 \\ -1, & x < 0 \\ 0, & x = 0 \end{cases}$$

Note: Kenall's τ should only be used for "monotonic" functions. Monotonic functions can only have an upward trend that is never downward, or vice versa. (Non-decreasing or non-increasing.)

Another Note: There are three commonly used versions of Kendall's τ. This one is known as Tau-a. Tau-b should be used for data where there ties.

3.5.1 Alternative Expression for Kendall's τ

We can express Kendall's τ in a more intuitive way. Remember a pair of observations is (x_i, y_i) and (x_j, y_j).

- Let n_c be the number of concordant pairs of observations.
- Let n_d be the number of discordant pairs of observations.
- Let N be the total number of possible unique pairs of observations.

$$\tau = \frac{n_c - n_d}{N}$$

or

$$\tau = p_c - p_d$$

where

- p_c is the proportion of times x and y increased together.
- p_d is the proportion of times y decreased when x increased.

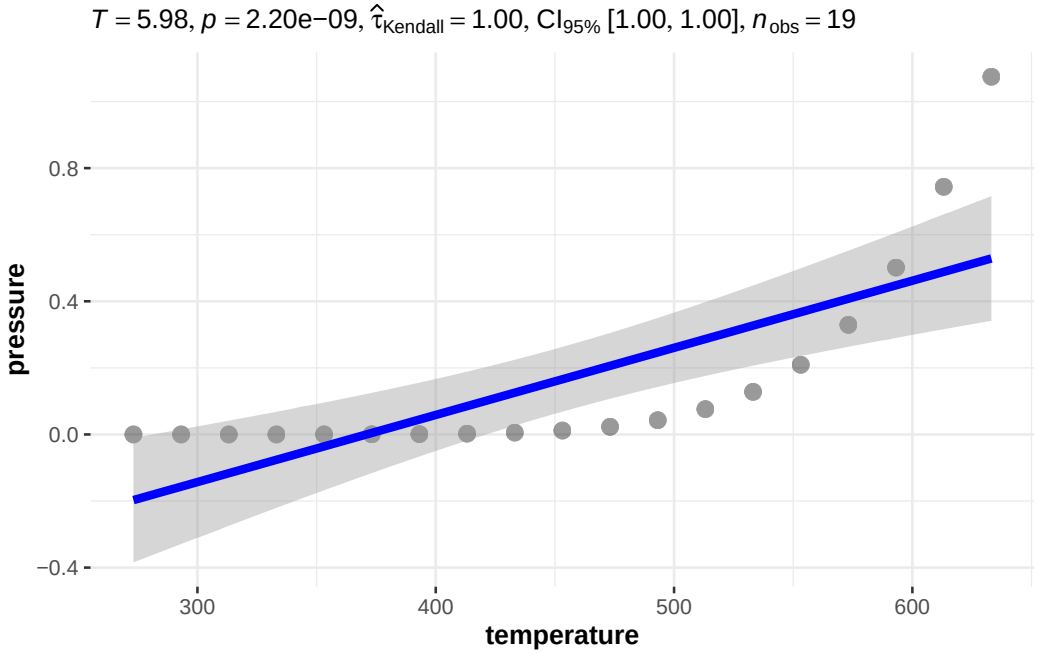
Note that $N = \binom{n}{2} = \frac{n(n-1)}{2}$, where n is the number of observations.

This may make the interpretation a little more graspable.

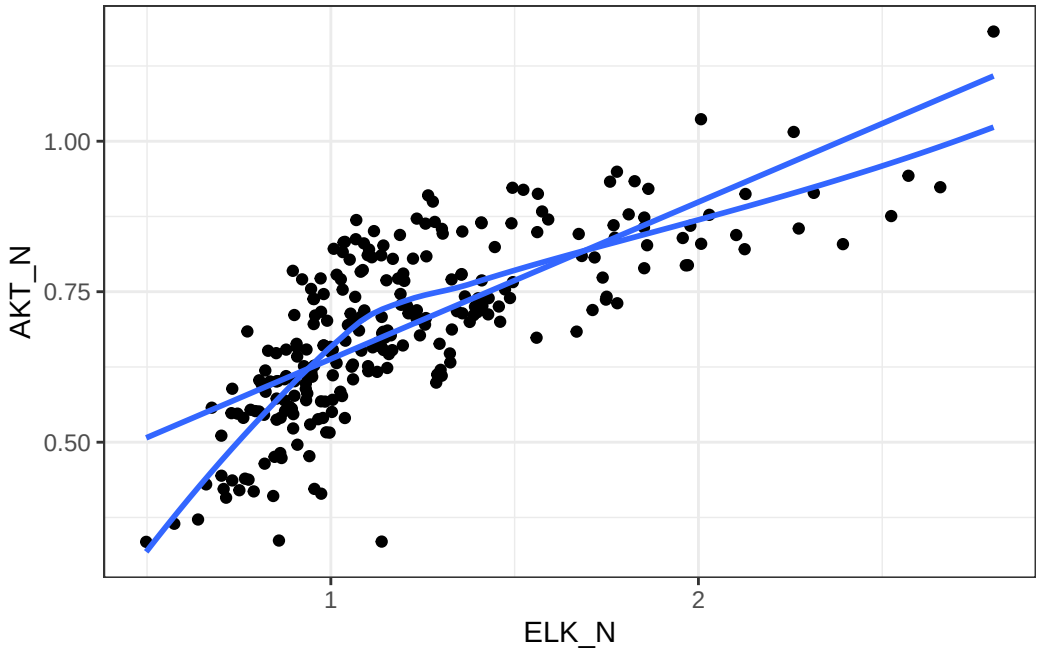
- $p_c + p_d$ must equal 1. (And $n_c + n_d$ must equal N). A consequence of this and the fact that $\tau = p_c - p_d$
  - $p_c = \frac{1+\tau}{2}$
  - $p_d = \frac{1-\tau}{2}$
- When τ is positive it is how much more often we see an concordance, i.e., x increases when y increases, compared to discordance where x increases and y decreases.
  - If $\tau = 0.5$ then $p_c = 0.75$ and $p_d = 0.25$. So 75% of the time y was increasing compared to 25% of the time
- When τ is negative its absolute value is how much more likely we are to see a decrease in y when x increases.
- If $\tau = -0.7$ then $p_c = 0.15$ and $p_d = 0.85$, so 85% of the time we saw a decrease in y whereas only 15% of the time we saw an increase in y

3.5.2 Kendall's τ with the pressure data

Recall the vapor pressure example.

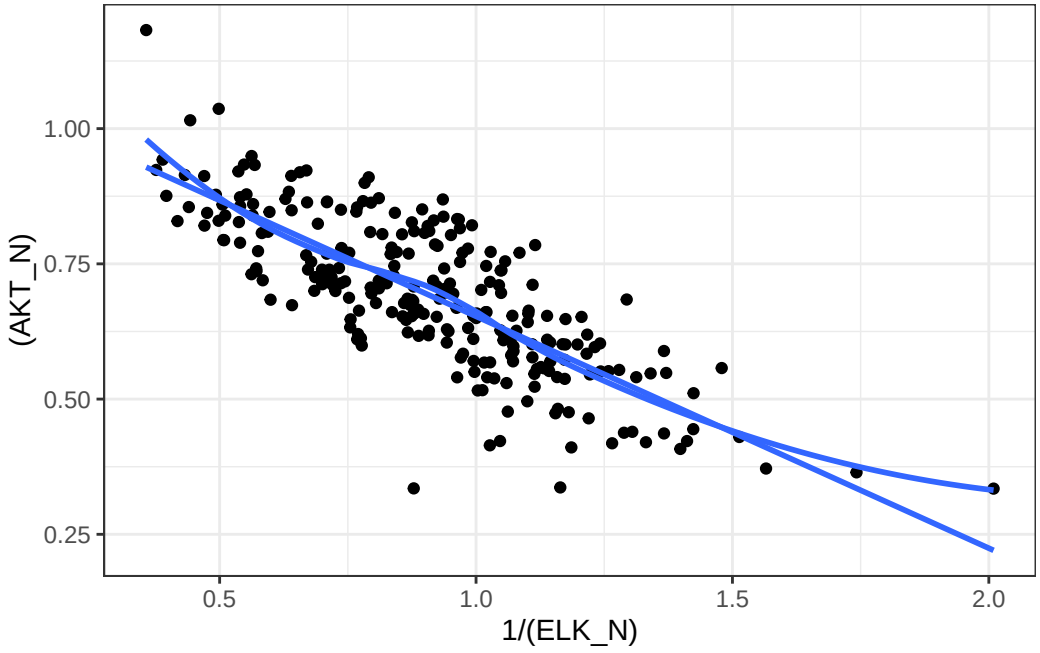


3.5.3 Kendall's τ on some mice proteins data



KendallCorr numeric	0.5866605
PearsonCorr numeric	0.7394402

3.5.4 A transformation



KendallCorr numeric	-0.5866605
PearsonCorr numeric	-0.7865069

3.5.4.1 Monotonic

- **Definition:** In the context of a sequence of numbers, "monotonic" means the sequence is either always increasing or always decreasing. It moves in one direction, without changing direction.
- **Explanation:** Think of a monotonic sequence like walking up or down a staircase. You're either consistently going upwards (increasing) or consistently going downwards (decreasing). You never switch directions and go up and then down, or down and then up.

Examples:

- **Monotonic Increasing:** 2, 5, 8, 11, 15 (each number is larger than the one before it)
- **Monotonic Decreasing:** 10, 7, 4, 1, -2 (each number is smaller than the one before it)

- **Not Monotonic:** 3, 6, 4, 8 (it increases, then decreases, so it's not monotonic)

Monotonic Transformation

- **Definition:** A monotonic transformation is a way of changing a set of numbers into a different set of numbers, but in a way that *preserves the order* of the original set.
- **Explanation:** Imagine you have a line of people arranged from shortest to tallest. A monotonic transformation would be like giving everyone in line platform shoes. Everyone gets taller, but the order from shortest to tallest stays the same.

Examples:

- **Original sequence:** 2, 5, 8
  - **Monotonic transformation (adding 3 to each number):** 5, 8, 11
  - **Monotonic transformation (multiplying each number by 2):** 4, 10, 16

Why is this important in statistics?

Monotonic transformations are useful in statistics because they can sometimes simplify data analysis *without changing the fundamental relationships* within the data. For instance, they can be used to:

- **Make data easier to work with:** Transforming data can sometimes make it easier to visualize or analyze.
- **Meet the assumptions of statistical tests:** Some statistical tests require data to have certain properties. Monotonic transformations can sometimes help data meet those assumptions.

Key takeaway: Monotonic means "always going in one direction." A monotonic transformation changes the values in a dataset but keeps the *order* the same.

4 Simple Linear Regression

```
knitr::opts_chunk$set(echo = TRUE, tidy = TRUE,
                      cache = T, message = FALSE, warning = FALSE)
# Very standard packages
library(tidyverse)
library(knitr)
library(ggpubr)
library(readr)

library(ggstatsplot)
# Globally changing the default ggplot theme.

## store default
old.theme <- theme_get()

## Change it to theme_bw(); i don't like the grey background.
## Look up other themes to find your favorite!
theme_set(theme_bw())
```

The Model, Estimating the Line, and Coefficient Inference

4.1 Statistical Models

Simple Model:

$$Y \sim N(\mu, \sigma)$$

This can be translated to:

$$Y = \mu + \epsilon$$

Where $\epsilon \sim N(0, \sigma)$.

This second form is the basis for **linear models**:

- The mean of a random variable Y can be written as a linear equation which in this case would be just a flat line.
- There is an error term ϵ that describes the deviation we expect to see from the mean.
- ϵ having a mean of 0 means that the linear equation for the mean of Y is correctly specified.
- By correctly specified I mean that we don't know what the actual mean of Y is, we are just crossing our fingers.

A more general model would be:

$$Y = g(x) + \epsilon$$

- $g(x)$ represents some sort of function with an argument x which outputs some constant.
- $g(x)$ is the **deterministic** portion of our model, i.e., it always gives the same output when given a single input.
- ϵ is the **random** component of our model.
- The whole entire point of statistics is that there is randomness and we have to figure out how to deal with it.
- We try to distill the signal $g(x)$ out of the noise ϵ of chaos/randomness. (Maybe overly "poetic")

Anyway, we try to take a stab at (guess) what the structure of $g()$ is.

$Y = \mu + \epsilon$ with $\epsilon \sim N(0, \sigma)$ is about as simple of a model as we get.

We will assume from hereon that $\epsilon \sim N(0, \sigma)$

4.1.1 The Linear Regression Model

We assume that the mean is some linear function of some variables x_1 through x_k.

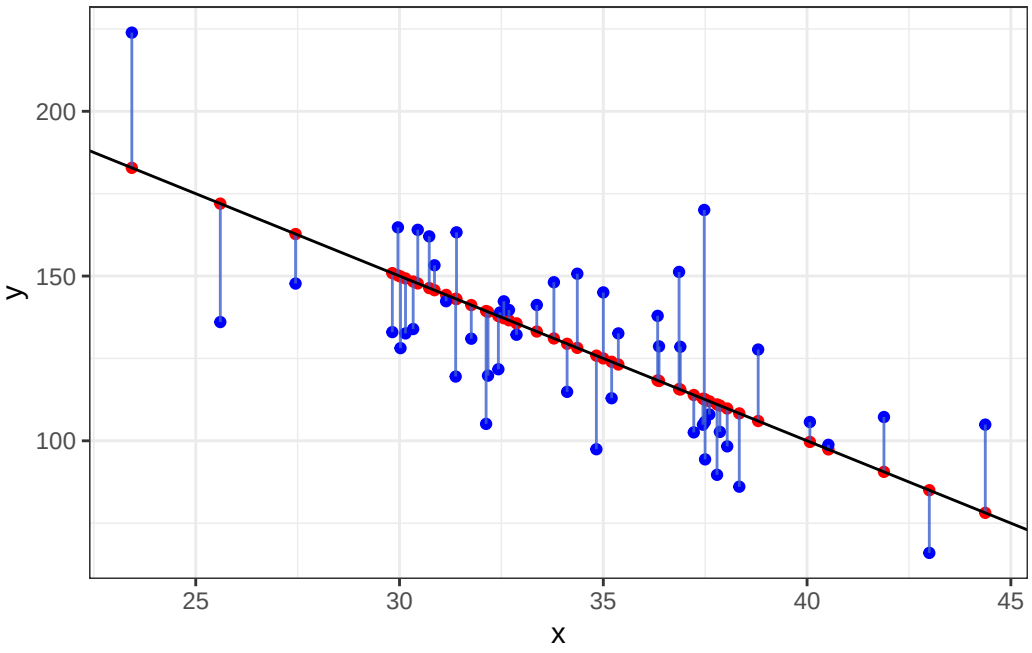
$$Y = \beta_0 + \sum_{i=1}^{k} \beta_i x_i + \epsilon$$

We will first start with the simplest linear regression model which is one that has only one input variable.

$$Y = \beta_0 + \beta_1 x + \epsilon$$

This gives us the form for how we could picture the data produced by the system we try to model.

```
n = 50

# Need x values
dat <- data.frame(x = rnorm(n, 33, 5))  #n, mu, sigma

# need a value for coefficients.
B <- c(300, -5)  # don't forget the y-intercept.

# Deterministic portion
dat$mu_y <- B[1] + B[2] * dat$x

# Introducing error
dat$err <- rnorm(n, 0, 20)

dat$y <- dat$mu_y + dat$err

ggplot(dat, aes(x = x, y = y)) + geom_point(data = dat, mapping = aes(x = x, y = mu_y),
    col = "red") + geom_point(data = dat, mapping = aes(x = x, y = y), col = "blue") +
    geom_segment(aes(xend = x, yend = mu_y), alpha = 0.8, col = "royalblue3") + geom_abline(
    slope = B[2]) + theme_bw()
```



- The black line is the true mean of Y at a given value of x: $\mu_{y|x} = 300 - 5x$
- The red dots are randomly chosen points along the line.
- The blue dots are what happens when we include that error term ϵ and shows how actual observations deviate from the line.
- Run this code a few times to see things sample to sample.

4.1.2 Comparing the Real to the Ideal

Here is data on 226 beers: ABV and Calories per 12oz.

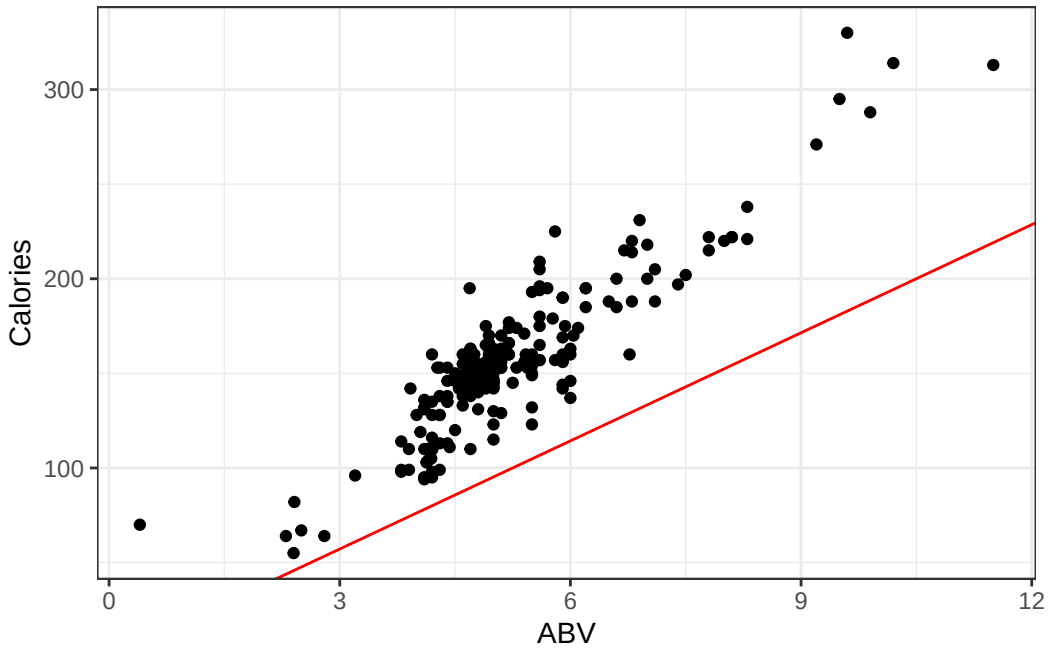
Alcohol "contains" calories, so the more alcohol in a beer means more calories!

Assuming you have a 12 ounce mixture of water and pure alcohol (ethanol) the exact equation for the number of calories based on ABV denoted by x is

$$f(x) = 19.05x + 0$$

Let's plot out the data on the beers and that equation.

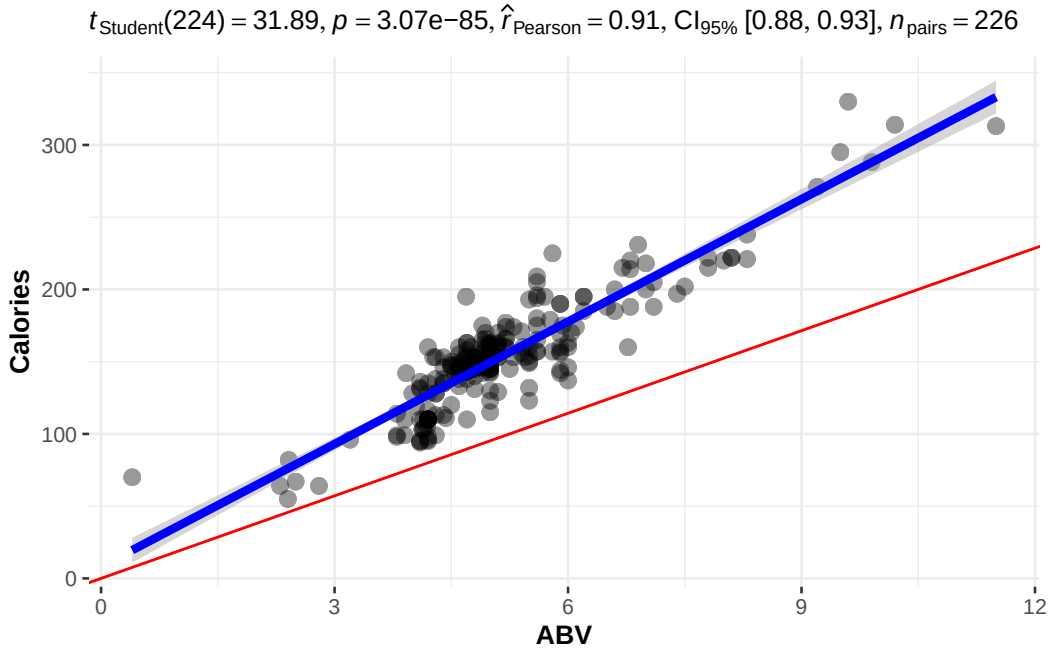
```
beer <- read.csv(here::here("datasets", "beer.csv"))

ggplot(beer, aes(x = ABV, y = Calories)) + geom_point() + geom_abline(slope = 19.05,
    col = "red")
```



Looks like that misses the mark. I suppose there is other stuff in beer besides alcohol.

So if were to plot a line for the actual on hand, perhaps you will agree that the one in blue is a "good" one.

```r
## OLD CODE ggplot(beer, aes(x = ABV, y = Calories)) + geom_smooth(method =
## 'lm', se = F) + geom_point() + geom_abline(slope = 19.05, col = 'red') +
## stat_cor()

# NEW CODE
ggscatterstats(data = beer, x = ABV, y = Calories, bf.message = FALSE, marginal = FALSE) +
    geom_abline(slope = 19.05, col = "red")
```



The blue line is the one we will learn how to make.

The line is meant to estimate the mean value of Y:

$$\mu_{y|x} = \beta_0 + \beta_1 x$$

We have to figure out what values we should use for β_0 and β_1.

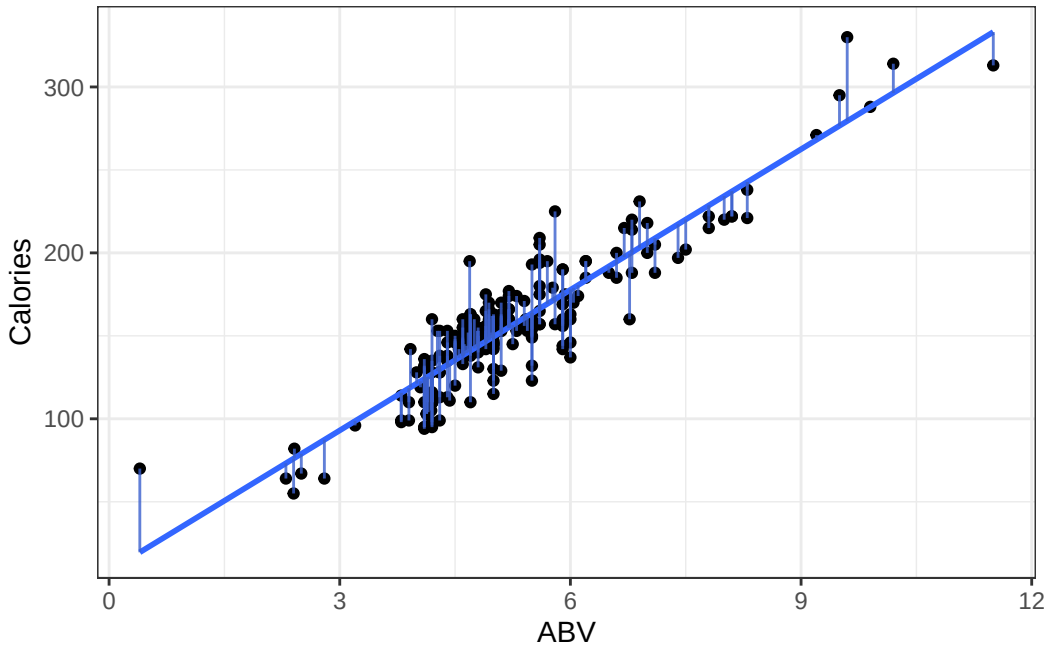
- β_0 and β_1 are the true and exact values for the intercept and slope of the line. These are unknown and are referred to as *parameters* of our model.
- We but $\widehat{\ }$ over the symbol for parameters to denote an estimate of the parameter.
- $\hat{\beta}_0$ is our y intercept estimate.
- $\hat{\beta}_i$ is our slope estimate.
- Our overall estimate for the line is $\hat{\mu}_{y|x} = \hat{\beta}_0 + \hat{\beta}_i x$
- Often we also write $\hat{y} = \hat{\beta}_0 + \hat{\beta}_i x$.

Regardless, what approach(es) can we take to create *good* values for $\hat{\beta}_0$ and $\hat{\beta}_i$?

4.2 Least Squares Regression Line

Here is a plot of the data with what is called the least squares regression line.

```
beerLm <- lm(Calories ~ ABV, beer)


ggplot(beer, aes(x = ABV, y = Calories)) + geom_point() + geom_smooth(method = "lm",
    se = FALSE) + geom_segment(aes(x = ABV, y = Calories, xend = ABV, yend = beerLm$fitted.va
    alpha = 0.8, col = "royalblue3") + theme_bw()
```



4.2.1 Measuring Error

- **S**um of **S**quared **E**rror

$$SSE = \sum_{i=1}^{n} \left(y_i - \left(\hat{\beta} + \hat{\beta} x_i\right)\right)^2 = \sum_{i=1}^{n} e_i^2$$

4.2.2 OLS solution (You can ignore this if you want.)

We want to find values of $\hat{\beta}_0$ and $\hat{\beta}_i$ that minimize the **error sum of squares**:

$$\begin{aligned} SSE &= \sum (y - \hat{y})^2 \\ &= \sum_{i=1}^{n} \left(y_i - \left(\hat{\beta}_0 + \hat{\beta}_i x_i \right) \right)^2 . \end{aligned}$$

To estimate to find

$$\frac{\partial}{\partial \hat{\beta}_i} SSE = \sum -2x(y - (\hat{\beta}_0 + \hat{\beta}_i)) = 0$$

After substituting in our equation for $\hat{\beta}_i$ above and doing quite a lot of algebra, we can make $\hat{\beta}_i$ the subject.

$$\begin{aligned} \hat{\beta}_i &= \frac{\sum xy - n\bar{x}\bar{y}}{\sum x^2 - n\bar{x}^2} \\ &= \frac{\sum (x - \bar{x})(y - \bar{y})}{\sum (x - \bar{x})^2} \end{aligned}$$

$$\begin{aligned} \frac{\partial}{\partial \hat{\beta}_0} SSE &= \sum -2 \left(y_i - \left(\hat{\beta}_0 + \hat{\beta}_i x_i \right) \right) = 0 \\ \sum \left(y_i - \left(\hat{\beta}_0 + \hat{\beta}_i x_i \right) \right) &= 0 \\ \sum y &= \sum \left(y_i - \left(\hat{\beta}_0 + \hat{\beta}_i x_i \right) \right) \\ \frac{\sum y}{n} &= \frac{\sum \left(\hat{\beta}_0 + \hat{\beta}_i x_i \right)}{n} \\ \bar{y} &= \hat{\beta}_0 + \hat{\beta}_i \bar{x} \end{aligned}$$

Substituting $\bar{x}$ into the regression line gives $\bar{y}$. In other words, the regression line goes through the point of averages.

4.2.3 Now you "know" the theory, lets look at what we do.

Okay, we're using `R`. That function does regression for us? `lm()` !!!

Use `?lm` to get a somewhat understandable summary that works pretty well once you learn how people screw up summarizing functions...

Any... Back to `lm()`

- there is a formula argument and a data argument. That's all you need to know for now.

Let's use it on that beers dataset and see what we get.

```
### Make an lm object

beers.lm <- lm(Calories ~ ABV, beer)

summary(beers.lm)
```

```
Call:
lm(formula = Calories ~ ABV, data = beer)

Residuals:
    Min      1Q  Median      3Q     Max
-40.738 -13.952   1.692  10.848  54.268

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   8.2464     4.7262   1.745   0.0824 .
ABV          28.2485     0.8857  31.893   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 17.48 on 224 degrees of freedom
Multiple R-squared:  0.8195,    Adjusted R-squared:  0.8187
F-statistic:  1017 on 1 and 224 DF,  p-value: < 2.2e-16
```

Our estimated regression line is:

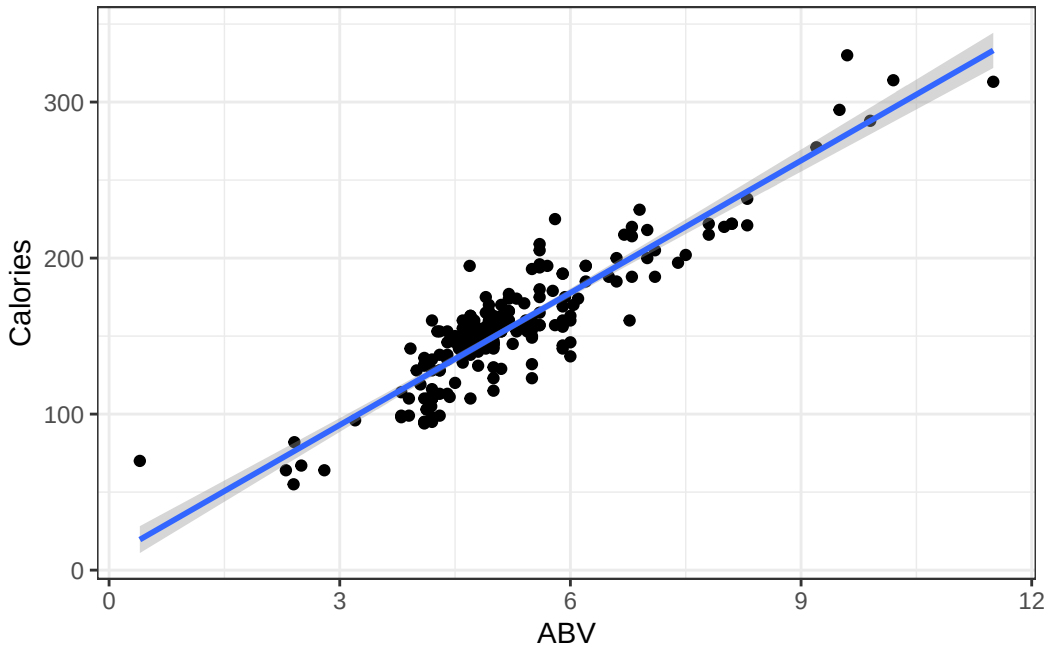
$$\hat{y} = 8.2364 + 28.2485x$$

Or to write it as an estimated conditional mean.

$$\hat{\mu}_{y|x} = 8.2364 + 28.2485x$$

And that's where we get this line:

```
ggplot(beer, aes(x = ABV, y = Calories)) + geom_point() + geom_smooth(method = "lm",
    se = TRUE)
```



Additionally for the $\epsilon \sim N(0, \sigma)$ term in the theoretical model,

- `Residual standard error: 17.48` in our output tells us
- $\hat{\sigma} = 17.48$ which means that at a given point on the line, we should expect the calorie content of individuals beers to fall above or below the line by about 17.48 calories.

4.2.4 Interpreting Coefficients

So we've got the slope estimate $\widehat{\beta}_i$ and we've got the intercept estimate $\widehat{\beta}_0$.

In general:

- The slope tells us how much we expect the mean of our y variable to change when the x variable increases by 1 unit.
- The intercept tells us what we would expect the mean value of y to be when the x variable is at 0 units.
- Sometimes the intercept does not make sense.
  - This is usually because $x = 0$ is not within the range of our data.
  - Or impossible.
  - Or near the $x = 0$ range, the relationship between

For our data:

- The intercept of 8.2464 indicates that the model predicts that the mean calorie content of "beers" with 0% ABV is 8.2464 calories.
  - There are some beers that are advertised as "non-alcoholic".
  - They have "negligble" amounts of alcohol but it is non-zero.
  - Whether this is completely sensical or not depends on what you define as a beer.
- The slope of 28.2485 indicates that when we look at the mean calorie of beers, it should be 28.2485 calories higher of beers with a 1% higher ABV.
  - This makes sense, i.e., alcohol has calories so more alcohol means more calories.

You always should look at your results and ask your self, "Does this make sense"?

- Usually the question of y-intercept making sense or not is not really a useful one.
- We need a y-intercept to have a line, and very often the data and what is reasonably observable does not include observations where $x \approx 0$.

4.2.5 Prediction Using The Line

For our beers data, for beers with 5% ABV, we would expect an average calorie content of

$$\hat{\mu}_{y|x} = 8.2364 + 28.2485 \cdot 5$$

$= 149.4789$

Great, but what if you tell me you make beer at home and measure the ABV to be 5%.

- How many calories are in that specific Beer?
- We could say approximately 150 calories but that let's not forget that we have an estimated error standard deviation of $\hat{\sigma} = 17.48$ so it might be better to say the calorie content is some where in the 131.9989 to 166.9589 range (if we're using a bunch of decimal places.)
- Honestly when doing casual intreptations it might be better to say 150 ± 20 because its all about imprecision anyway.

4.2.6 Predict Function in R

Use the `predict()` function to get estimates for the mean which can be point estimates/predictions.

- The point of a regression model is estimate some sort model, and the method for doing so is called `predict`. It works like so:

```
predict(lmModel, newdata)
```

- Here `lmModel` is an already estimated model, and `newdata` is a dataset (referred to as data frame in R) containing the new cases, real or imaginary, for which we want to make predictions.

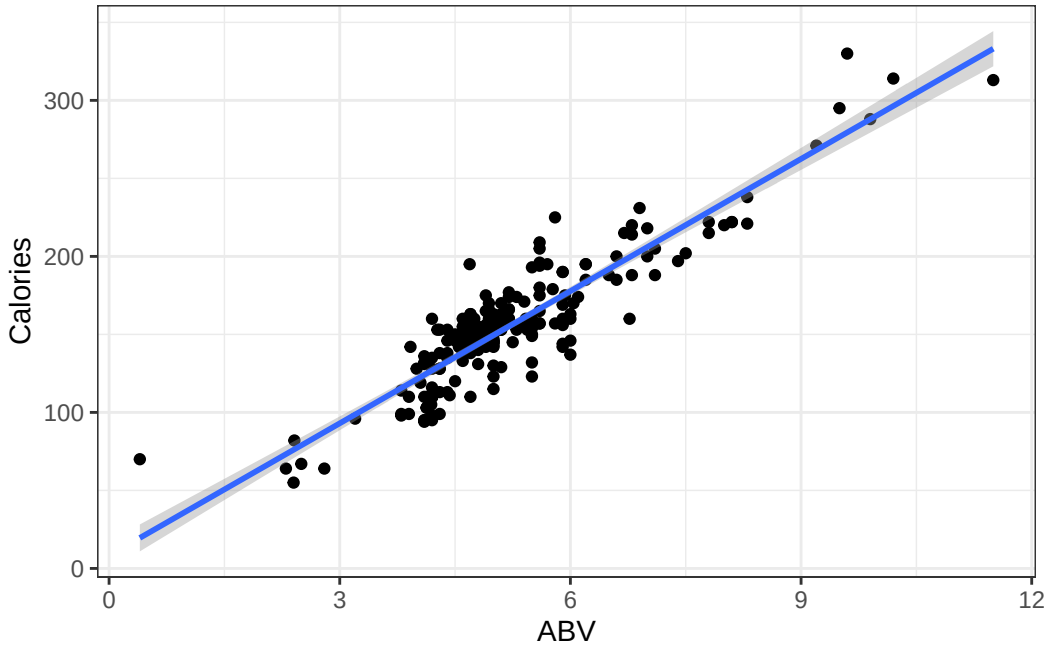
```
# Create a dataset for predicting average calories in beers with ABVs of 1%,
# 2%, 3%, ..., 10% R can create this vector easily with the `:` operator.
# `A:B` creates a vector that starts at A and goes up by 1 until it reaches
# (but does not exceed) B 1:10 represents the vector c(1, 2, 3, 4, 5, 6, 7, 8,
# 9, 10)

predictionData = data.frame(ABV = 1:10)


predict(beers.lm, newdata = predictionData)
```

```
       1         2         3         4         5         6         7         8
 36.49494  64.74349  92.99203 121.24058 149.48913 177.73768 205.98623 234.23478
       9        10
262.48333 290.73188
```

- If you do not correctly specify the correct vector name in your `newdata` argument, then `predict` will simply give you the fitted $\hat{y}$ values for each point in your data.

4.3 Statistcal Inference in Linear Regression

```
ggplot(beer, aes(x = ABV, y = Calories)) + geom_point() + geom_smooth(method = "lm",
    se = TRUE)
```



There's bands around that line.

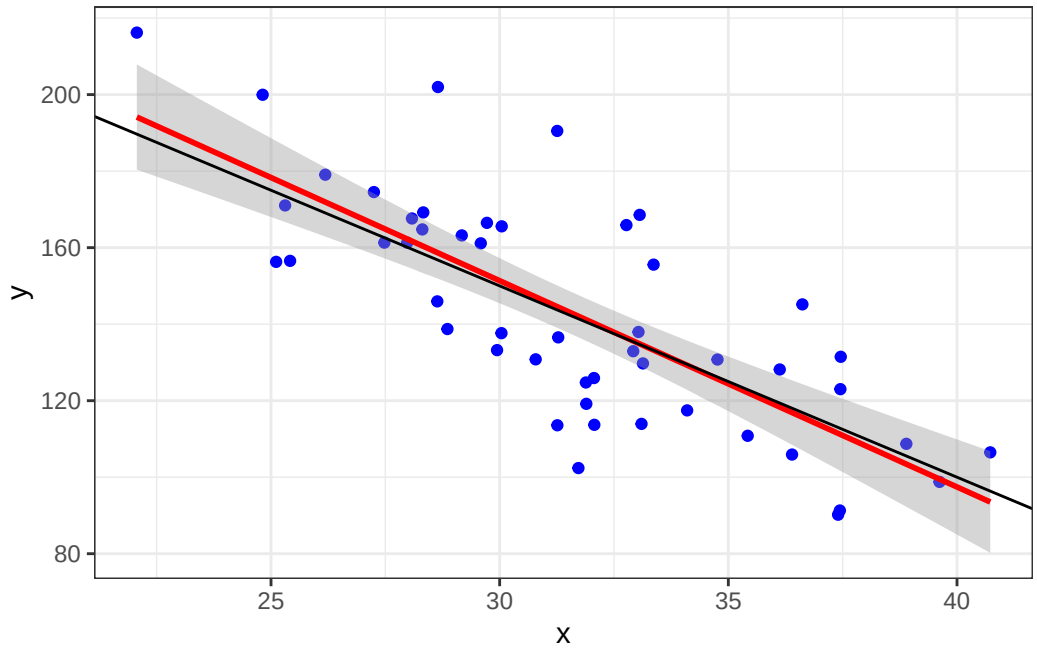
- This represents the fact that we don't know the true regression line and are trying to account for how imprecise our estimate is.
- It displays a continuum for plausible values of the true line $\hat{\mu}_{y|x}$ based on our sample data.

4.3.1 Example

Let's look at what happens when the true model is $Y = 300 - 5x + \epsilon$ when $\epsilon \sim N(0, 20)$ and we take a sample of 50 individuals.

- The sample observations will be plotted in blue.
- The estimated regression line from those observations will be red.
- The true line will be in black.

```
n = 50

set.seed(11)

# Need x values
dat <- data.frame(x = rnorm(n, 33, 5))  #n, mu, sigma

# need a value for coefficients.
B <- c(300, -5)  # don't forget the y-intercept.

# Deterministic portion
dat$mu_y <- B[1] + B[2] * dat$x

# Introducing error
dat$err <- rnorm(n, 0, 20)

dat$y <- dat$mu_y + dat$err

ggplot(dat, aes(x = x, y = y)) + geom_point(data = dat, mapping = aes(x = x, y = y),
    col = "blue") + geom_smooth(method = "lm", col = "red") + geom_abline(intercept = B[1],
    slope = B[2], col = "black") + theme_bw()
```



Given that we have sample data, we can't expect our line to be the true line.

This is because there is variability associated with our slope estimate and variability associated with our intercept estimate.

4.3.2 Tests for the Line Coefficients

The $\widehat{\beta}$'s are referred to as coefficients.

When you see the output for the summary of our `lm`, draw your attention to this part of the output.

```
    Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   8.2464     4.7262   1.745   0.0824 .
ABV          28.2485     0.8857  31.893   <2e-16 ***
```

We are seeing various pieces of information. From left to right:

- We point estimates for the $\widehat{beta}$'s.
- We are given standard errors for those estimates.
- We are given test statistics foir some hypothesis test (what is it)?
- We are given the p-value for that test.

The hypothesis test is this:

$$H_0 : \beta_i = 0$$
$$H_1 : \beta_i \neq 0$$

The test statistic is:

$$t = \frac{\widehat{\beta}_i}{SE_{\widehat{\beta}_i}}$$

We simple divide our coefficient estimate by its standard error.

The test statistic is assumed to be from a t distribution with degrees of freedom $df = n - 2$.

For the y-intercept, we get a p-value of 0.0824:

- Our conclusion would be that there is moderate evidence that there is a true y-intercept in the model.
- Inference on the y-intercept is considered not important usually.

For the slope the p-value is approximately 0:

- Conclusion: There is extremely strong evidence of a linear component that relates ABV with calories.
- This is usually what we are concerned with.

4.3.3 Confidence Intervals

We can likewise get confidence intervals for the coefficients. They take the general form:

$$\hat{\beta}_i \pm t_{\alpha/2, n-2} \cdot SE_{\hat{\beta}_i}$$

Again the $t_{\alpha/2}$ quantile depends on the value for α of the desired nominal confidence level $1 - \alpha$.

To get the confidence intervals we use the `confint()` function.

```
confint(beers.lm, level = 0.99)
```

```
                0.5 %   99.5 %
(Intercept) -4.031982 20.52476
ABV         25.947491 30.54961
```

Our interval indicates **statistically** plausible (which means disregarding context) levels for a y-intercept of the true line is between -4.03 and 20.52 with 99% confidence.

And plausible values for the increase in calories when beers have 1% higher ABV is between 25.95 and 30.55 with 99% confidence.

Do these CIs indicate any compatibility with the line for the theoretical line that computes (exactly) the amount of calories in a can of water with percentage of pure ethanol by volume?

$$f(x) = 19.05x$$

5 Inference on the Regression Line

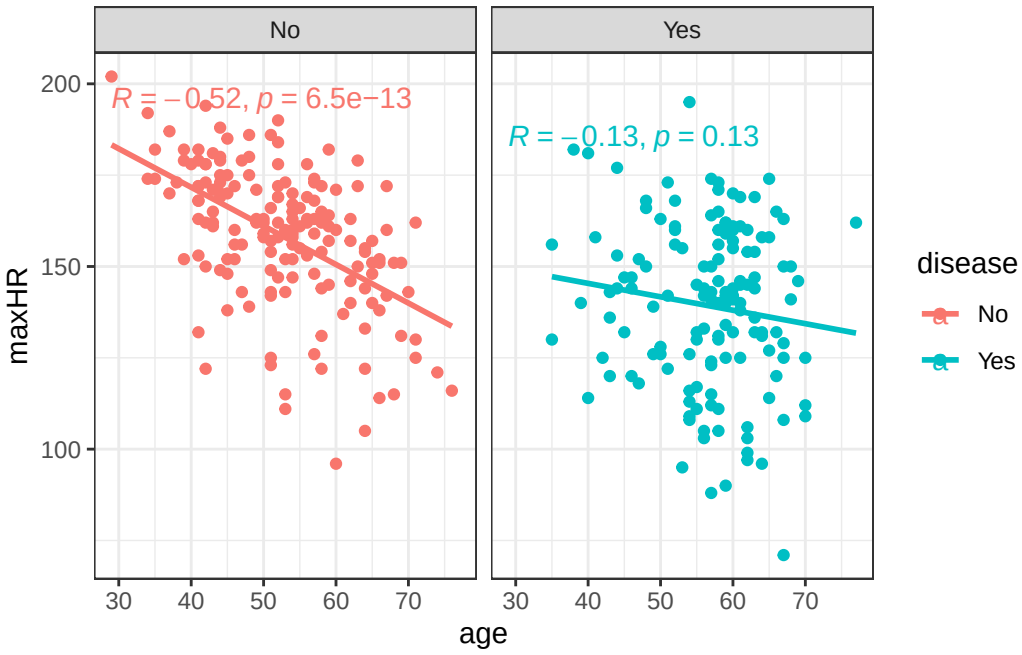
Here are some code chunks that setup this document.

```
# Here are the libraries I used
library(tidyverse)
library(knitr)
library(readr)
library(ggpubr)
```

```
# This changes the default theme of ggplot
old.theme <- theme_get()
theme_set(theme_bw())
```

```
# Fun fact. If you put the dataset in the same folder as the Rmd/qmd file you are
# working in. Then you only need the filename to load it. Not the whole file
# path on your computer.
#
# I prefer to use the here package to resolve local path issues

heart <- readr::read_csv(
  here::here("datasets","Heart.csv")
                  )
```

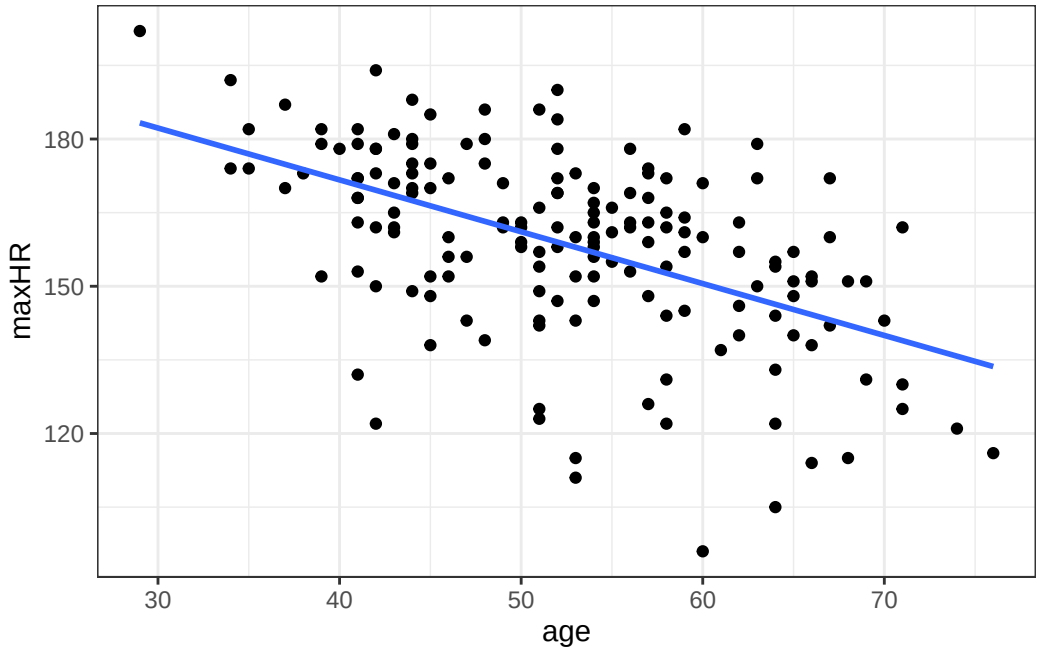
```
ggplot(heart, aes(x = age, y = maxHR, color = disease)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  facet_wrap(~disease) +
  stat_cor()
```



We can `filter()` the data and only examine individuals without heart disease since the relation doesn't appear viable in those with heart disease.

```
noDisease <- filter(heart, disease == "No")
```

```
ggplot(noDisease, aes(x = age, y = maxHR)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE)
```



Anyway, there's a line let's see what the equation for it is.

```
# nD is short for no Disease
nD.lm <- lm(maxHR ~ age, noDisease)

nD.lm.sum <- summary(nD.lm)

nD.lm.sum
```

```
Call:
lm(formula = maxHR ~ age, data = noDisease)

Residuals:
    Min      1Q  Median      3Q     Max
-54.545  -7.782   2.524  10.004  31.624

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 213.9282     7.2204  29.628  < 2e-16 ***
age          -1.0564     0.1351  -7.818 6.47e-13 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 16.41 on 162 degrees of freedom
Multiple R-squared:  0.2739,    Adjusted R-squared:  0.2694
F-statistic: 61.12 on 1 and 162 DF,  p-value: 6.469e-13
```

So the equation to our line would be:

$\widehat{m}axHR = 213.9 - 1.06 \cdot age$

5.0.0.1 Confidence Intervals for Coefficients

```
confint(nD.lm, level = 0.99)
```

```
                0.5 %      99.5 %
(Intercept) 195.108145 232.7482574
age          -1.408595  -0.7041663
```

5.1 Uncertainty in the Model

$$y = \beta_0 + \beta_1 x + \epsilon_{error}$$

$$\epsilon \sim N(\theta, \sigma_\epsilon)$$

The ϵ representing the inherent variability we would expect from individual observations.

5.2 Partitioning Variability

At this point, we will be examining data for those without heart disease only, unless mentioned otherwise.

$$\hat{y} = \hat{\mu}_{y|x} = \hat{\beta}_0 + \hat{\beta}_1 x$$

5.2.1 Sums of Squares

$$SSTO = \Sigma_{i=1}^{n} (y_i - \bar{y})^2$$

$$s_y^2 = \frac{SSTO}{n-1}$$

This variability in the variable can be broken up into two pieces:

1. The **Sum of Squares of Regression** SSR.

$$SSR = \Sigma(\hat{y} - \bar{y})^2$$

2. The **Sum of Squared Error** SSE.

$SSE = \Sigma(y_i - \hat{y}_i)^2$

The two sums of squares combined are the total variability $SSTO$.

$SSTO = SSR + SSE$

5.2.2 Coefficient of Determination R^2

$R^2 = \frac{SSR}{SSTO} = \frac{Explained}{Total}$

which is the ratio of how much of the total variability in our data that is "explained" by our model.

Often the form of R^2 is written as

$R^2 = 1 - \frac{SSE}{SSTO}$

5.2.2.1 Interpreting R^2

"The regression model is accounting for $<R^2 \cdot 100>\%$ of the variability in <the response variable."

```
nD.lm.sum
```

```
Call:
lm(formula = maxHR ~ age, data = noDisease)

Residuals:
    Min      1Q  Median      3Q     Max
-54.545  -7.782   2.524  10.004  31.624

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 213.9282     7.2204  29.628  < 2e-16 ***
age          -1.0564     0.1351  -7.818 6.47e-13 ***
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 16.41 on 162 degrees of freedom
Multiple R-squared:  0.2739,    Adjusted R-squared:  0.2694
F-statistic: 61.12 on 1 and 162 DF,  p-value: 6.469e-13
```

Regardless of how we get it, an interpretation of the model would be "using a linear model, age accounts for approximately 27% of the variability in maximum heart rate".

5.3 Analysis of Variance (ANOVA) in Regression

In simple linear regression we discussed the hypothesis test for the slope (and intercept).

$$H_0 : \beta_1 = 0$$

- Essentially, null hypothesis posits that x variable is useless as predictor of y

$$H_1 : \beta_1 \neq 0$$

In essence we are testing to see if the x variable is a worthwhile predictor.

This test was done via the test statistic

$$t = \frac{\hat{\beta}_1}{SE_{\hat{\beta}_1}}.$$

The t distribution with $n-2$ **degrees of freedom** is used to compute the p-value for this hypothesis test.

5.3.1 Degrees of Freedom

- SSR has 1 degree of freedom.
- SSE has $n-2$ degrees of freedom.
- Overall, $SSTO$ has $n-1$ degrees of freedom.

5.3.2 Mean Squares and the Test Statistic

However, in regression analysis, we are mainly interested in the **mean square of regression** MSR and **mean squared error**.

A mean square is the sum of square divided by its degrees of freedom.

- $MSR = \frac{SSR}{1}$

- $MSE = SSE \div (n-2)$

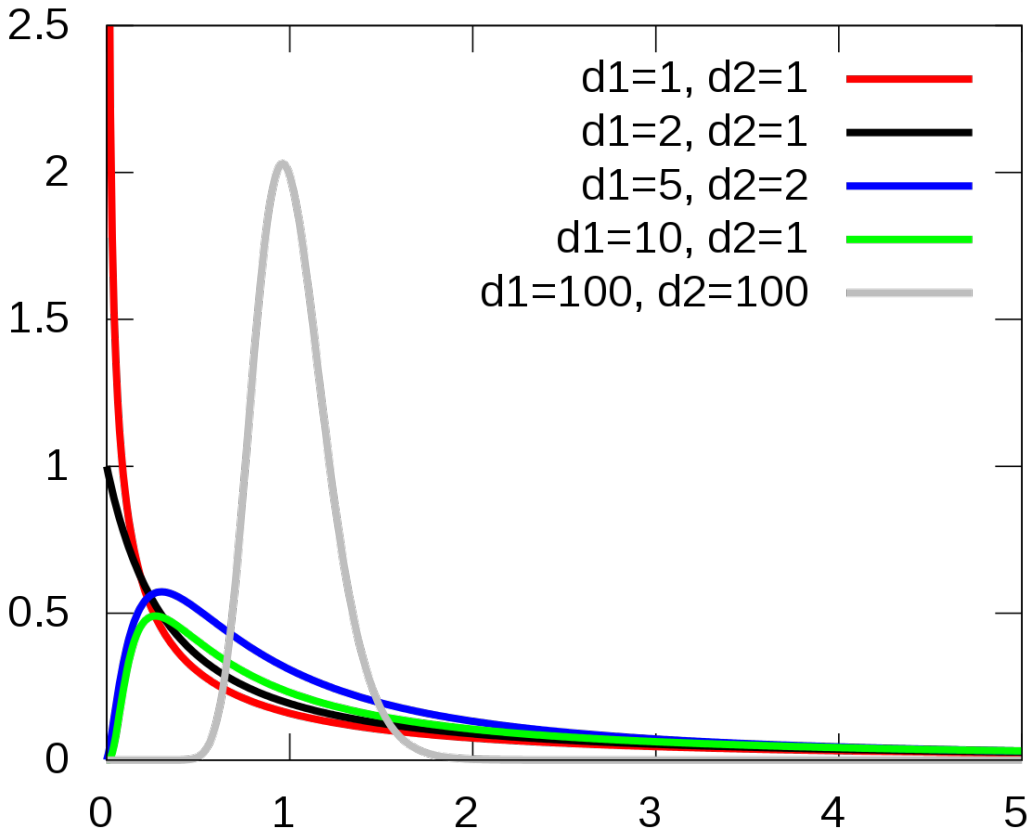
We take their ratio, which is our new *test statistic*.

$F_t = \frac{MSR}{MSE} = \frac{SSR}{1} \div \frac{SSE}{n-2}$

F-test under H_0

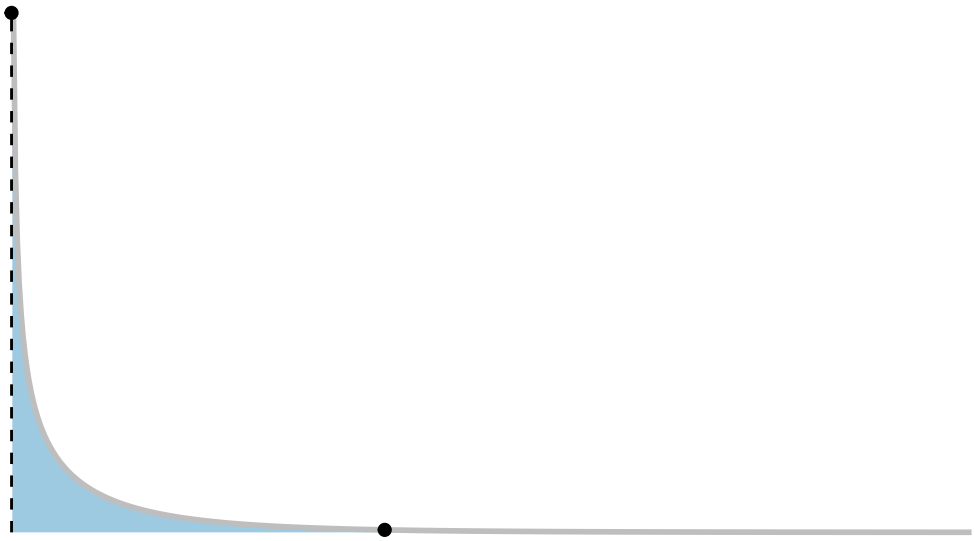
$$H_0 \to F_t \sim F(1, n-2)$$

5.3.3 F-Distribution



The p-value is the probability of a random value from an $F(1, n-2)$ distribution exceeding the test statistic: $p = P\left(F(1, n-2) > F_t\right)$.

```r
library(ggdist)
library(distributional)

dist_df = data.frame(
  d = dist_f(1,18)
)

dist_df %>%
  ggplot(aes(xdist = d)) +
  stat_slab(color = "grey", expand = TRUE,
            aes(fill = after_stat(level)),
            .width = c(.95,1))+
  stat_spike(at = function(x) hdci(x, .width = .975),
             linetype = "dashed")+
  # need shared thickness scale so that stat_slab and geom_spike line up
  scale_thickness_shared() +
  theme_void() +
    scale_fill_brewer(na.value = "gray95") +
  labs(fill = "Rejection Region on F-distribution",
       caption = "F distribution with 1 and 18 degrees of freedom") +
  theme(legend.position = "none")
```



F distribution with 1 and 18 degrees of freedom

5.3.4 ANOVA Table

We would in general summarise a Analysis of Variance based hypothesis test with the following ANOVA table.

Source of Variability	Sum of Squares	df	MS	F	p-value
Regression/Model	SSR	1	$MSR = SSR/1$	$F_t = MSR/MSE$	p
Error	SSE	$n-2$	$SSE/(n-2)$		
Total	$SSTO$	$n-1$			

5.3.5 Regression ANOVA in R

The ANOVA hypothesis test information is shown in the last row of the `summary` of an `lm`.

```
nD.lm.sum
```

```
Call:
lm(formula = maxHR ~ age, data = noDisease)

Residuals:
    Min      1Q  Median      3Q     Max
-54.545  -7.782   2.524  10.004  31.624

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 213.9282     7.2204  29.628  < 2e-16 ***
age          -1.0564     0.1351  -7.818 6.47e-13 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 16.41 on 162 degrees of freedom
Multiple R-squared:  0.2739,    Adjusted R-squared:  0.2694
F-statistic: 61.12 on 1 and 162 DF,  p-value: 6.469e-13
```

```
anova(nD.lm)
```

```
Analysis of Variance Table

Response: maxHR
           Df Sum Sq Mean Sq F value    Pr(>F)
age         1  16458 16457.7  61.115 6.469e-13 ***
Residuals 162  43625   269.3
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

5.4 Model Error: σ_ϵ

This is the linear model for each value y value in a random sample:

$y_i = \beta_0 + \beta_1 x_i + \epsilon_i$

Where of particular note, we have

variability:

$$\epsilon_i \sim N(0, \sigma_\epsilon)$$

$$\hat{\sigma}_\epsilon = \sqrt{MSE}$$

5.4.1 Standard Error of $\hat{\beta}_1$ and $\hat{\beta}_0$

$$SE_{\beta_0} = \hat{\sigma}_\epsilon \sqrt{\frac{1}{n} + \frac{\overline{x}^2}{\sum (x_i - \overline{x})^2}}$$

$$SE_{\beta_1} = \hat{\sigma}_\epsilon \sqrt{\frac{1}{\sum (x_i - \overline{x})^2}}$$

5.4.2 Standard Error for the Line

We use the equation

$$\hat{y} = \hat{\mu}_{y|x} = \beta_0 + \beta_1 x$$

How uncertain are we estimating the mean, or making predictions?

5.4.2.1 Estimated Conditional Mean

When estimating $\mu_{y|x}$ at a given value of x, the standard error is:

$$SE_{\hat{\mu}_{y|x}} = \hat{\sigma}_\epsilon \sqrt{\frac{1}{n} + \frac{(x - \overline{x})^2}{\sum (x_i - \overline{x})^2}}$$

This is the counterpart to what you did in your introductory course when just estimating the mean of y (μ) without considering an x variable.

$$SE_{\bar{y}} = \frac{s}{\sqrt{n}}$$

5.4.2.2 Standard Error of Predictions

$$SE_{pred} = \hat{\sigma}_\epsilon \sqrt{1 + \frac{1}{n} + \frac{(x - \overline{x})^2}{\sum (x_i - \overline{x})^2}}$$

5.4.3 Confidence Intervals for the Mean

We can use our data and compute an estimated value for the line:

$$\hat{\mu}_{y|x} = \beta_0 + \beta_1 x$$

We can create confidence intervals for this estimate using the t-distributiuon with $n-2$ degrees of freedom.

$$\hat{\mu}_{y|x} + t_{\alpha/2,df} \cdot SE_{\hat{\mu}_{y|x}} = \hat{\mu}_{y|x} \pm t_{\alpha/2,n-2} \hat{\sigma}_\epsilon \sqrt{\frac{1}{n} + \frac{(x - \overline{x})^2}{\sum (x_i - \overline{x})^2}}$$

This interval tells us that we can be $(1 - \alpha)100\%$ confident that the true line at a given point x is between the lower and upper bound.

5.4.4 Prediction Intervals for Future Observations

In a similar manner, we can create confidence intervals that will let us be $(1-\alpha)100\%$ confident that a future observation will be between

$$\hat{y} \pm t_{\alpha/2,n-2} SE_{pred} = \hat{y} \pm t_{\alpha/2,n-2} \hat{\sigma}_\epsilon \sqrt{1 + \frac{1}{n} + \frac{(x - \overline{x})^2}{\sum (x_i - \overline{x})^2}}$$

5.4.5 Getting Confidence and Prediction Intervals in R

For confidence and prediction intervals, we can use the `predict()`. The arguments we need to define are `interval` and `level`.

- `interval` can be set to "none", "confidence", and "prediction" for no interval, a confidence interval for the mean, and a future observation prediction interval, respectively
- `level` is simply the confidence level of your interval. The default is 0.95.

```
# need the data for predictions

predData <- data.frame(age = c(20, 30, 40, 50, 60, 70, 80))

confIntervals <- predict(nD.lm, newdata = predData, interval = "confidence", level = 0.99)
predIntervals <- predict(nD.lm, newdata = predData, interval = "prediction", level = 0.99)
```

Confidence Intervals:

```
confIntervals
```

```
       fit      lwr      upr
1 192.8006 180.8474 204.7537
2 182.2368 173.6092 190.8644
3 171.6730 166.1228 177.2232
4 161.1092 157.6473 164.5711
5 150.5454 146.3056 154.7852
6 139.9816 132.9975 146.9657
7 129.4178 119.2006 139.6349
```

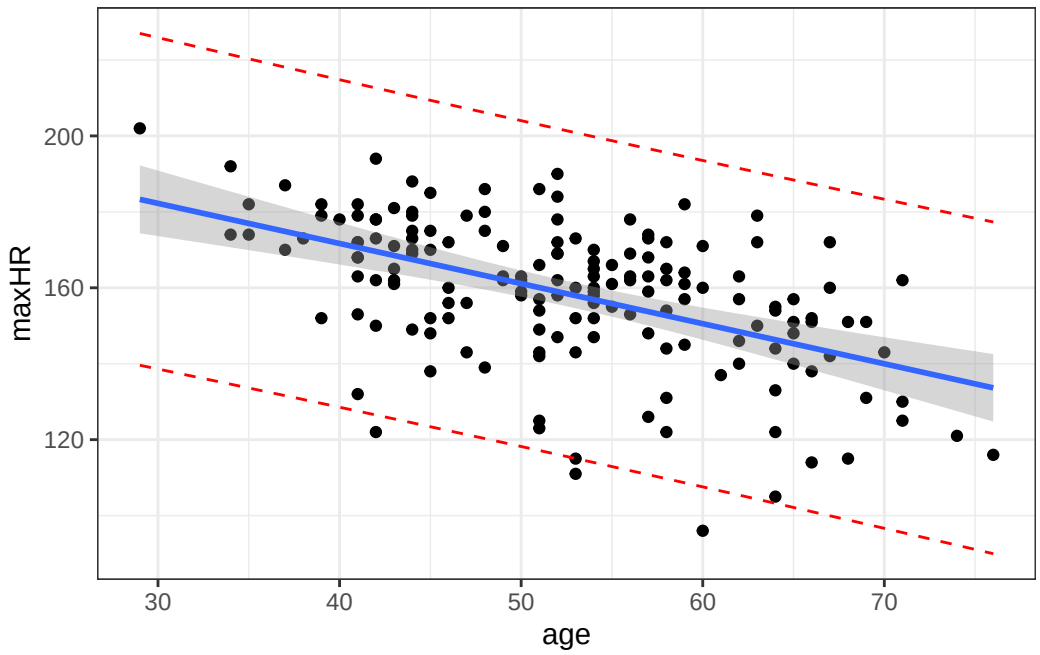
Prediction Intervals:

```
predIntervals
```

```
       fit       lwr      upr
1 192.8006 148.38873 237.2125
2 182.2368 138.60227 225.8713
3 171.6730 128.54132 214.8046
4 161.1092 118.19624 204.0221
5 150.5454 107.56269 193.5281
6 139.9816  96.64206 183.3211
7 129.4178  85.44134 173.3942
```

5.4.6 Graph of Confidence Intervals and Prediction Intervals

```
# without giving predict() new data,
# you get predictions for ALL points in the data
predictions <- predict(nD.lm,
                       interval = "prediction",
                       level = 0.99)

# combin prediction intervals with dataset
allData <- cbind(noDisease, predictions)

# graph
 #define x and y axis variables
ggplot(allData, aes(x = age, y = maxHR)) +
  geom_point() + #add scatterplot points
  stat_smooth(method = lm, level = 0.99) + #confidence bands
  geom_line(aes(y = lwr), col = "red",
            linetype = "dashed") + #lwr pred interval
  geom_line(aes(y = upr), col = "red",
            linetype = "dashed") #upr pred interval
```



Blue is estimated line, gray is possibilities for true line, and red is a range possible future

observations.

5.4.7 Important Note: Confidence Levels and Their Reliability.

The confidence level for a confidence/prediction interval only applies to that individual interval.

5.5 Working-Hotelling Confidence:

The calculation of confidence and prediction bands if fairly simple. You just have to replace the critical t value used to create the confidence and prediction interval. This replacement is based off of a modification of values from the F-Distribution.

This value will be referred to is as W_α. It requires a value from the F distribution with 2 degrees of freedom for the numerator, and $n-2$ for the denominator. This value on the F distribution is such that it has a left-tail area/probability of $1-\alpha$.

$$W_\alpha = \sqrt{2 \cdot F(1-\alpha, 2, n-2)}$$

5.5.1 Working-Hotelling Confidence Bands

To compute the confidence bands, the formula is:

$$\hat{y} \pm t_{\alpha/2,n-2} \cdot SE_{pred} = \hat{y} \pm W_\alpha \sqrt{1 + \frac{1}{n} + \frac{(x-\overline{x})^2}{\sum (x_i - \overline{x})^2}}$$

5.5.2 Working-Hotelling Prediction Bands Bands

Similarly for the prediction intervals.

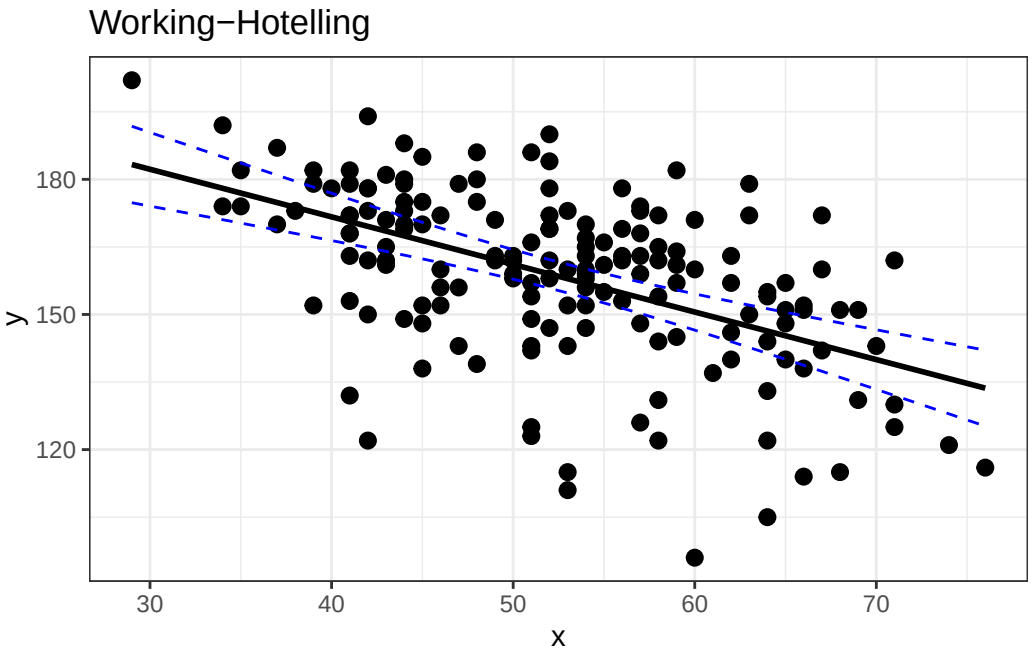
$$\hat{y} \pm t_{\alpha/2,n-2} SE_{pred} = \hat{y} \pm W_\alpha \hat{\sigma}_\epsilon \sqrt{1 + \frac{1}{n} + \frac{(x-\overline{x})^2}{\sum (x_i - \overline{x})^2}}$$

5.5.3 Getting These in R

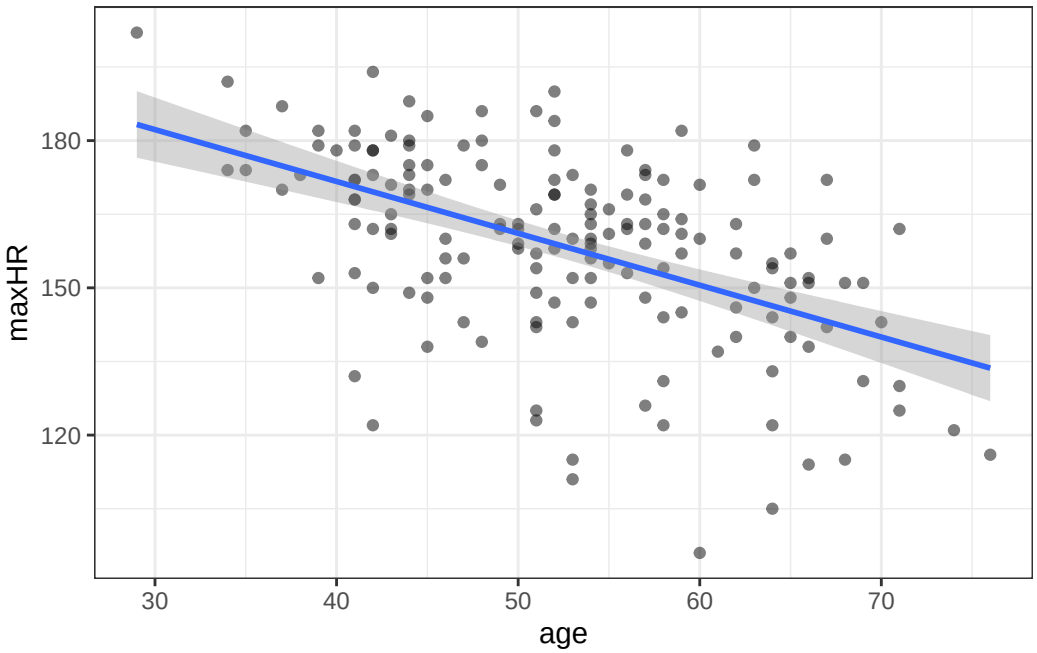
Apparently I need to make my own R functions for this.

```
working_hotelling_intervals <- function(x, y) {
  y <- as.matrix(y)
  x <- as.matrix(x)
  n <- length(y)

  # Get the fitted values of the linear model
  fit <- lm(y ~ x)
  fit <- fit$fitted.values

  # Find standard error as defined above
  se <- sqrt(sum((y - fit)^2) / (n - 2)) *
    sqrt(1 / n + (x - mean(x))^2 /
           sum((x - mean(x))^2))

  # Calculate B and W statistics for both procedures.
  W <- sqrt(2 * qf(p = 0.95, df1 = 2, df2 = n - 2))
  B <- 1-qt(.95/(2 * 3), n - 1)

  # Compute the simultaneous confidence intervals

  # Working-Hotelling
  wh.upper <- fit + W * se
  wh.lower <- fit - W * se

  xy <- data.frame(cbind(x,y))

  # Plot the Working-Hotelling intervals
  wh <- ggplot(xy, aes(x=x, y=y)) +
    geom_point(size=2.5) +
    geom_line(aes(y=fit, x=x), size=1) +
    geom_line(aes(x=x, y=wh.upper),
              colour='blue', linetype='dashed') +
    geom_line(aes(x=x, wh.lower),
              colour='blue', linetype='dashed') +
    labs(title='Working-Hotelling')


  return(wh)
}
```

```
working_hotelling_intervals(x = noDisease$age, y = noDisease$maxHR)
```




```
# compare with ggplot
#
ggplot(noDisease,
       aes(x=age,
           y=maxHR)) +
  geom_point(alpha = .5) +
  geom_smooth(method = "lm",
              formula = y~x,
              se = TRUE)
```


maxHR
180
150
120
30
40
50
60
70
age

6 Residual Diagnostics

We will expand on these regression topics some when we move on to multiple regression.

Here are some code chunks that setup this document.

```
# Here are the libraries I used
library(tidyverse)
library(knitr)
library(readr)
library(ggpubr)
```

```
# This changes the default theme of ggplot
old.theme <- theme_get()
theme_set(theme_bw())
```

```
beer <- read.csv(here::here("datasets",
                            "beer.csv"))

beers.lm <- lm(Calories ~ ABV, beer)
```

6.1 Validating the Model and Statistical Inference: The Residuals

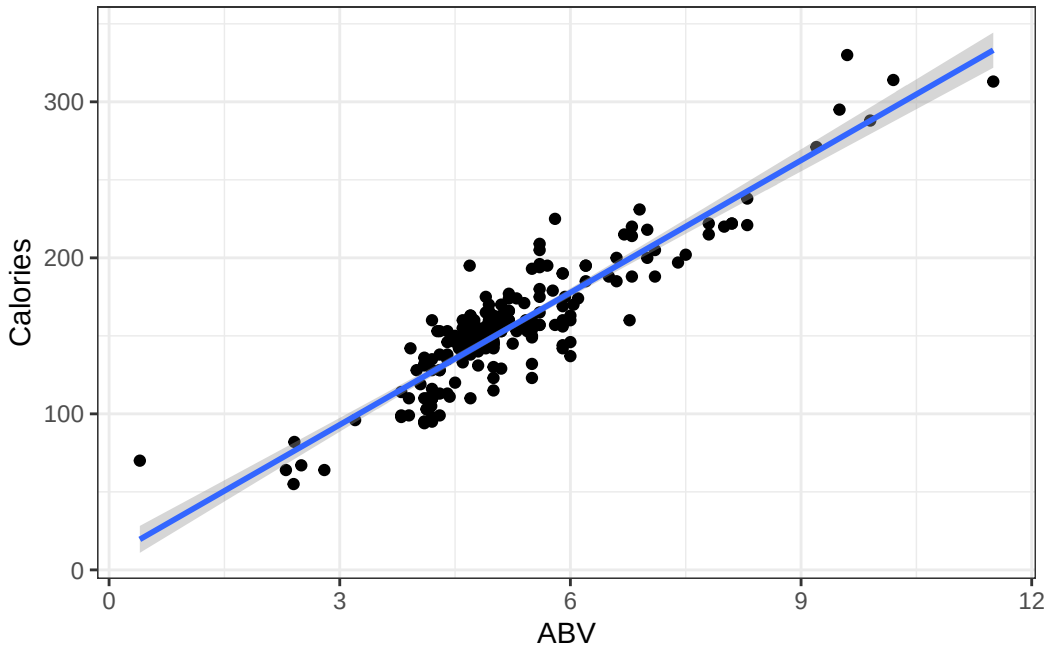
$$y = \beta_0 + \beta_1 x + \epsilon$$

$\epsilon \sim N(0, \sigma)$

There are assumptions that this model implies:

1. The error terms are normally distributed.
2. The error terms for all observation have a mean of zero which implies the model is unbiased,i.e, there is truly and only a linear relationship between x and y.
3. The error terms have the same/constant standard deviation/variability that does not depend on where we look at along the line. This is referred to as homogeneity of variance.
4. A required assumption is that the error terms of the observations are all independent of each other.

```
ggplot(beer,
       aes(x=ABV,y=Calories)) + geom_point() +
  geom_smooth(method = "lm",
              formula = y~x)
```

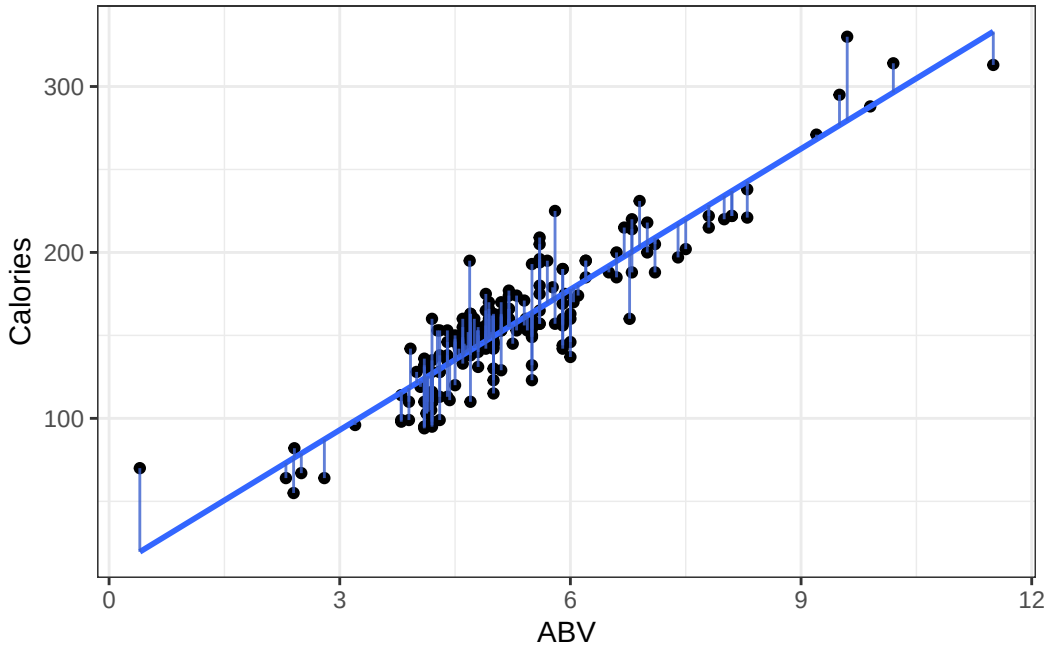


6.2 Residuals

We estimate the error using what are referred to as **residuals**:

$$e_i = y_i - \hat{y}_i = y_i - (\beta_0 + \beta_i x_i)$$

- y_i is observed value of y
- $\hat{y}_i$ is "fitted" value of y




```
summary(beers.lm)
```

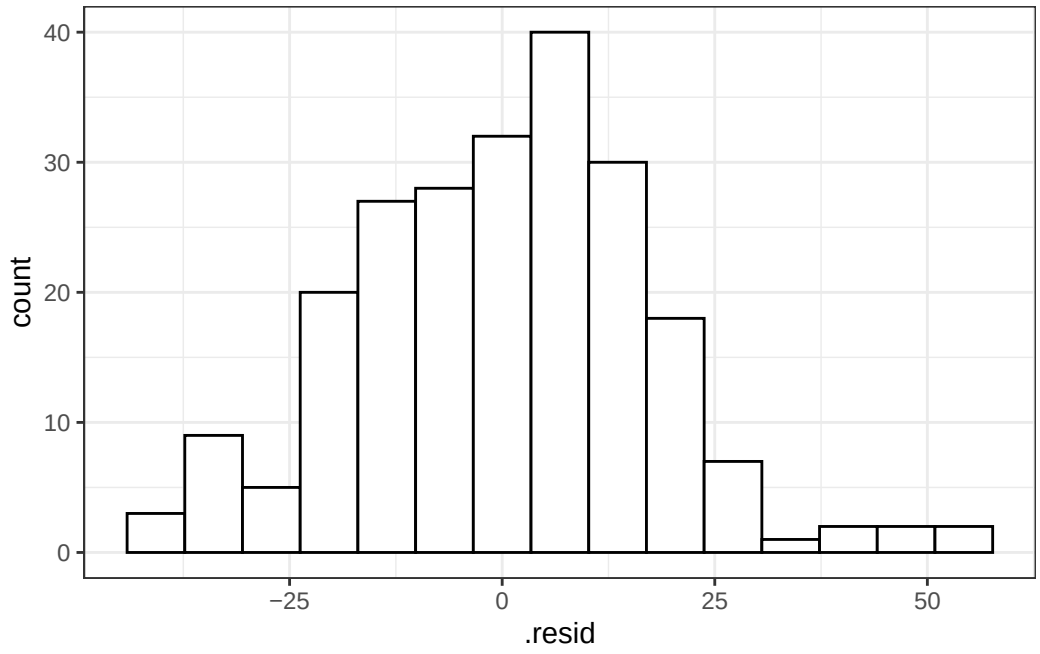
```
Call:
lm(formula = Calories ~ ABV, data = beer)

Residuals:
    Min      1Q  Median      3Q     Max
-40.738 -13.952   1.692  10.848  54.268

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   8.2464     4.7262   1.745   0.0824 .
```

```
ABV          28.2485     0.8857  31.893   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 17.48 on 224 degrees of freedom
Multiple R-squared:  0.8195,    Adjusted R-squared:  0.8187
F-statistic:  1017 on 1 and 224 DF,  p-value: < 2.2e-16
```

6.3 Checking Normality

You can use `beers.lm` in `ggplot()`. Use `.resid` for the variable you want to graph.

```
# The histogram function needs you
# to tell it how many bins you want.

ggplot(beers.lm, aes(x = .resid)) +
  geom_histogram(bins = 15,
                 color = "black", fill = "white")
```



6.3.1 QQ-Plots (QQ stands for QuantileQuantile)

Quantile-Quantile Plot or Probability Plot:

1. Order the residuals: $e_{(1)}, e_{(2)}, ..., e_{(n)}$

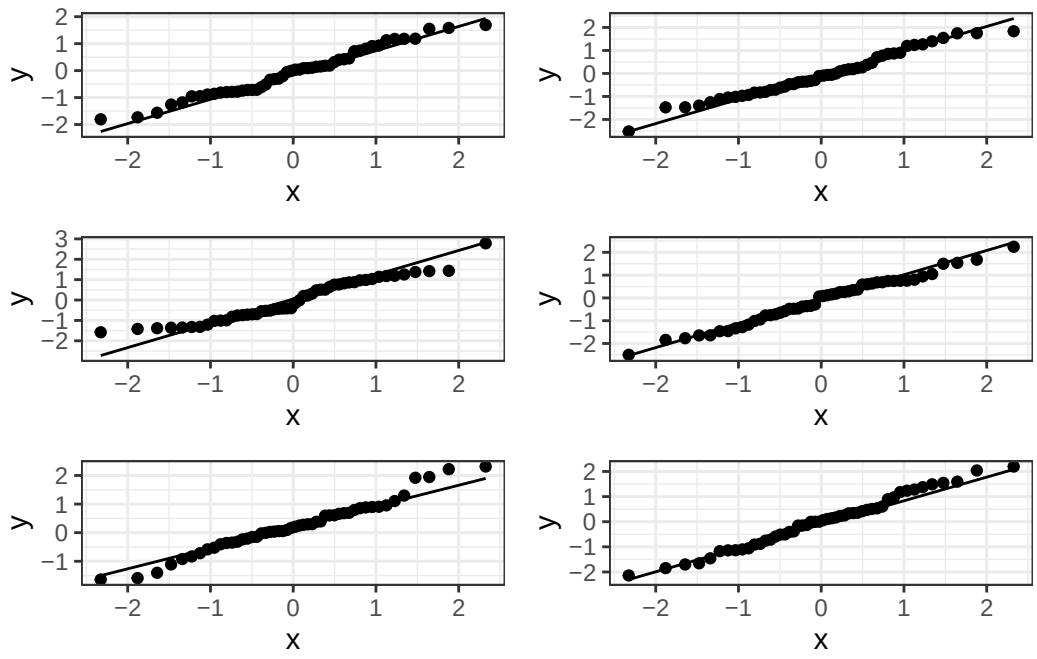
- The parentheses in the denominator indicate that the values are ordered from 1st to last in terms of least to greatest.
- $e_{(1)}$ is the minimum, $e_{(n)}$ is the maximum and then there is everything in between.

2. Find z values from the standard normal distribution that match the following:

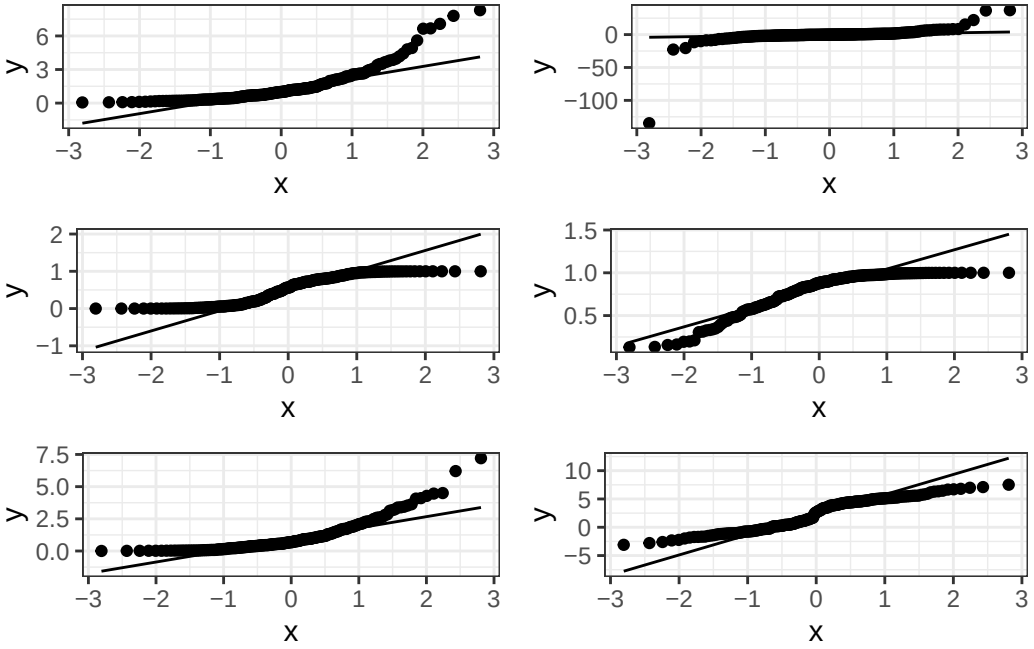
$$P(Z \leq z_{(i)}) = \frac{3i-1}{3n-1}$$

3. Plot the $e_{(i)}$ values against the $z_{(i)}$ values. You should see a straight line.

6.3.1.1 Good normal QQ-plot:

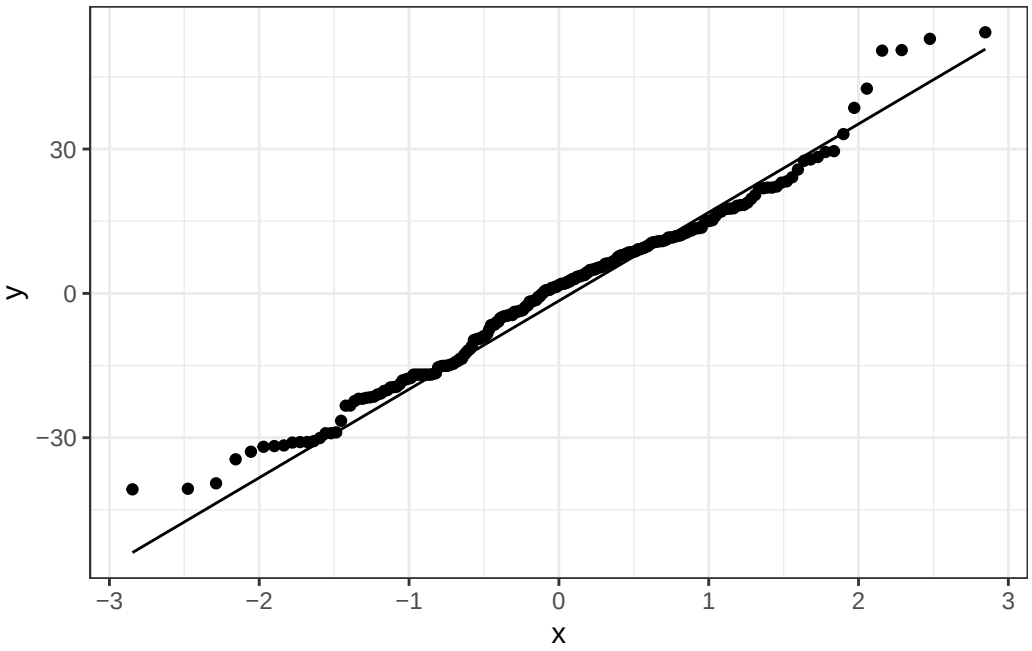


6.3.1.2 Bad plots:



6.3.1.3 Beer QQ Plot

```
ggplot(beers.lm, aes(sample = .resid)) +
  stat_qq() + stat_qq_line()
```



Things for the most part look good.

- There is some deviation at the tails, but that's expected.
- Essentially there is not much of a pattern to the deviation from the line except for the tails.
- This is honestly pretty ideal and somewhat rare for what we would see in the real world.

6.3.2 Hypothesis Tests for Normality

The hypotheses are:

H_0 : Data are normally distributed H_1 : Data are not normally distributed.

One such test is the Shapiro-Wilk Test which should only be used on *model residuals*

```r
shapiro.test(beers.lm$residuals)
```

```
    Shapiro-Wilk normality test

data:  beers.lm$residuals
W = 0.98379, p-value = 0.01097
```

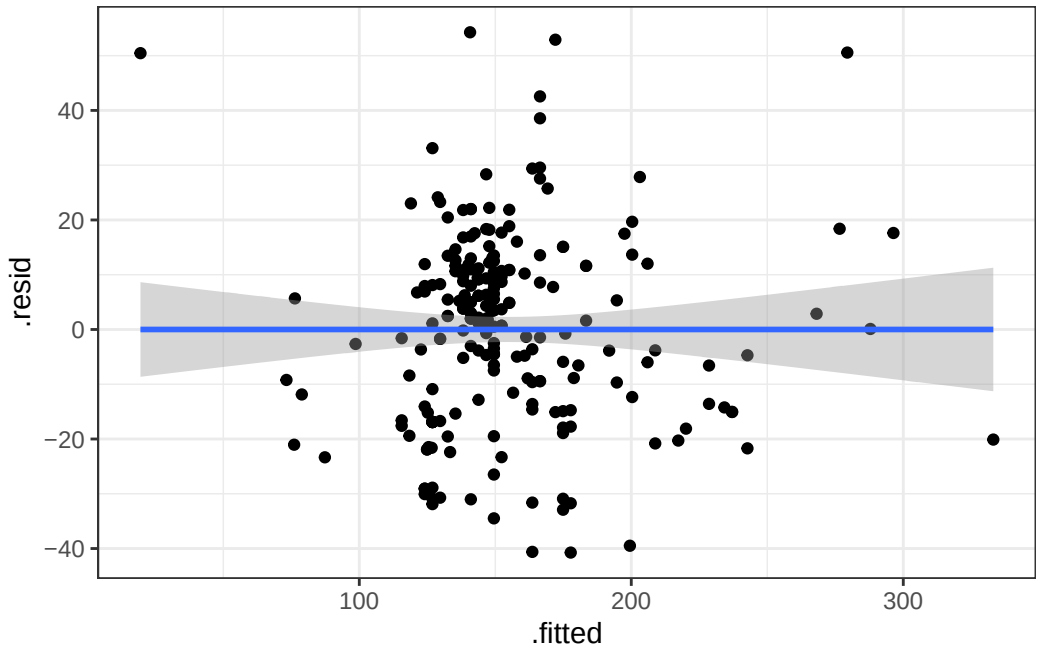
6.4 Residual Plots for Assessing Bias and Variance Homogeneity

Residual plots are where we plot the residuals on the vertical axis (y-axis) and the fitted values or observed x_i values from the data on the horizontal (x-axis).

We can use `beers.lm` in `ggplot()`.

- `.fitted` can be used for the fitted values variable.
- `.resid` can be used for the residuals values variable.

```
ggplot(beers.lm, aes(x = .fitted, y = .resid)) +
  geom_point() +
  geom_smooth(method = "lm")
```

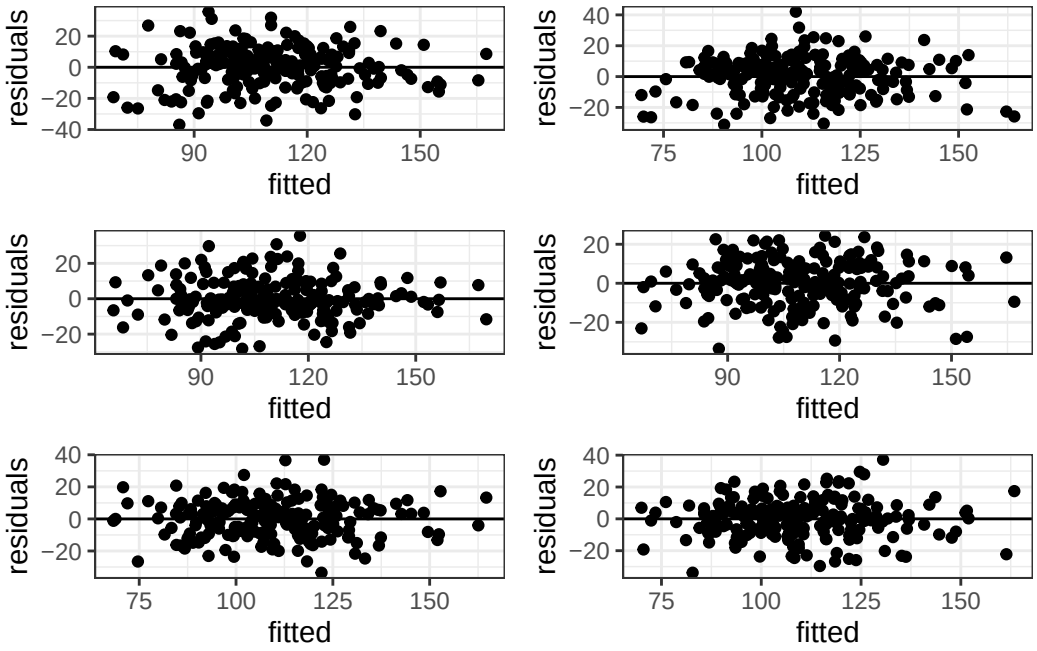


6.4.1 Premise of Residual Plots

- This is a way to assess whether the mean of the residuals is consistently zero across the regression line.
  - This is checking whether the model is biased or not.
  - If we see some sort of pattern where the residuals change direction, that means the data do not follow a linear pattern.

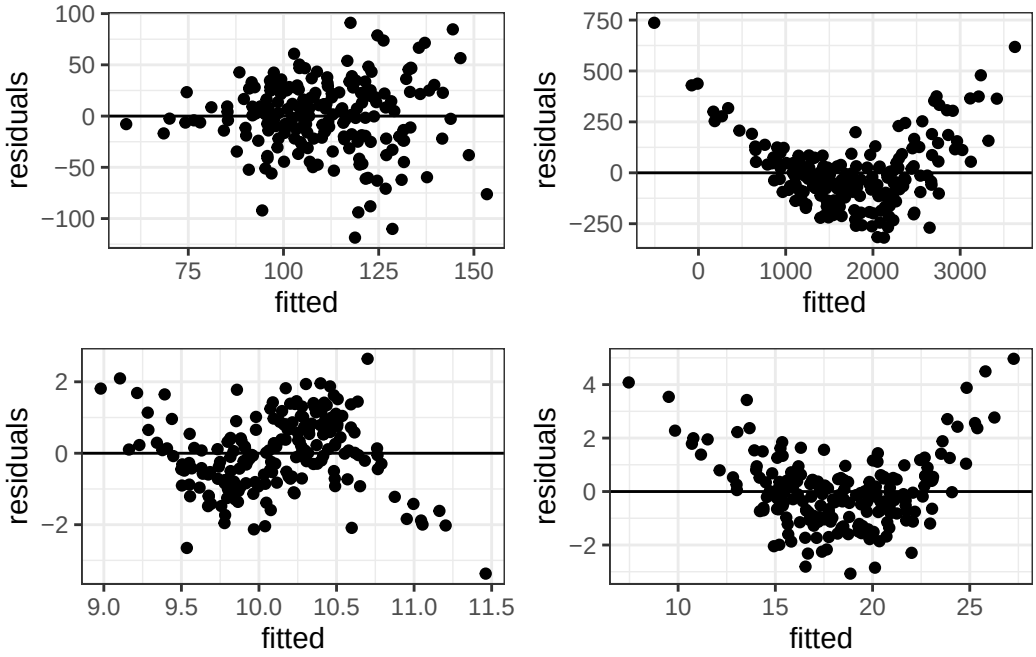
- It also lets us assess whether the variaibility is constant.
  - The residuals should form a pattern with equal vertical dispersion throughout.
  - If we see a pattern of increasing or decreasing spread, then the constant variance (homogeneity/homoskedasticity) assumption is violated.
- In general, you are look for randomly scattered points with no patterns whatsoever.
- The residual plot should form basically a circle or ellipse.

6.4.2 Good Residual Plots



The plots are centered at zero across there isn't much to say the variability is not constant across the whole plot.

6.4.3 Bad Residual Plots

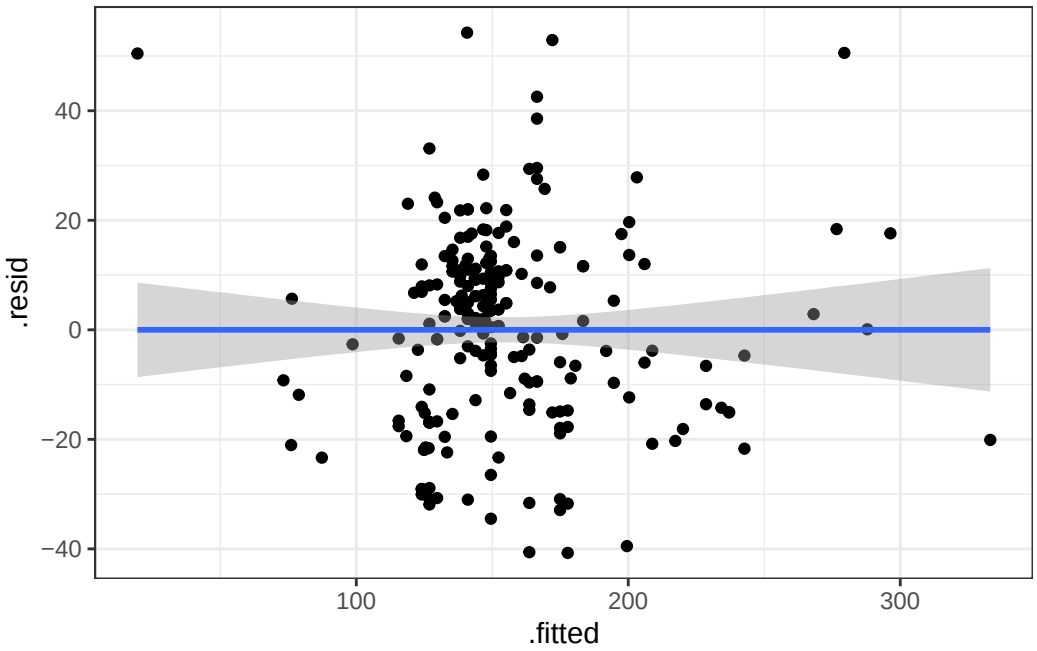


- Top left plot shows increasing variability but mean of 0. Unbiased and heterogeneous variance.
- Top right show increasing variability AND mean of not 0 consistently. Biased and heterogeneous variance.
- Bottom left shows the mean not being consistently 0. Biased but homogeneous variance.
- Bottom right shows a definite curvature and potential outliers.

6.4.3.1 Beer Data Residual Plots

Let's look at our beer data residual plots.

```
ggplot(beers.lm, aes(x = .fitted, y =.resid)) +
  geom_point() +
  geom_smooth(method = "lm")
```



This plot looks mostly fine. There are a few points that stick out on the far right and far left. The most peculiar one is in the top left.

It has the smallest fitted value, so let's pull that one out.

Calories	Beer	ABV
70	O'Doul's	0.4

O'Douls is a "non-alcoholic" beer. It may not be considered representative of our data. We could justify removing it from the data we are analyzing if our objective is analyze "alcoholic" beverages.

6.5 Outliers

Outliers are values that separate themselves from the rest of the data in some "significant way".

- How do we decide what is, and what is not an outlier.

standardized residuals:

$$z_i = \frac{e_i}{\hat{\sigma}_\epsilon} = \frac{y_i - \hat{y}}{\hat{\sigma}_\epsilon}$$

studentized residuals:

$$z_i^* = \frac{e_i}{\hat{\sigma}_\epsilon \sqrt{1 - h_i}}$$

h_i is a measure of "leverage". **Leverage** is a measure of how extreme a value is in terms of the predictor variable. It indicates the possibility that the outlier could strong influence the estimated regression line.

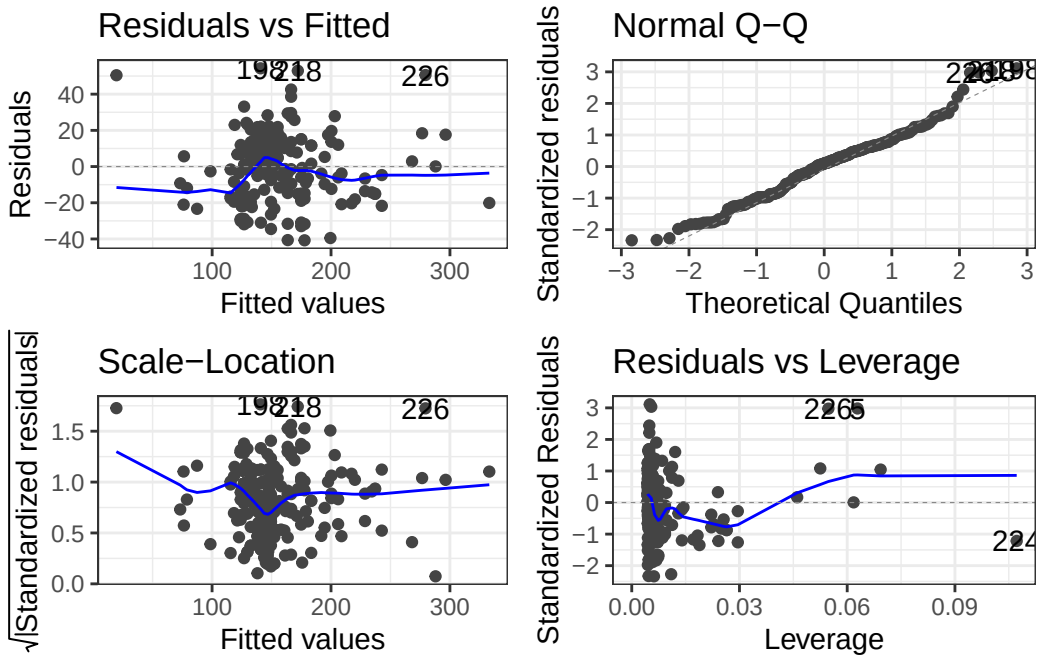
Sometimes, we use the square root of the standardized/studentized residuals, $\sqrt{|z_i|}$, to determine outliers. Then we would be looking for values that exceed $\sqrt{3} \approx 1.7$.

6.6 Alternative Way to Get Residual Diagnostics Graphs

There is a library called `ggfortify` which has a function that creates some useful plots. It is the `autoplot()` function.

This function only requires your `lm` model as an input to work. We'll do this on the original model.

```
library(ggfortify)

autoplot(beers.lm)
```



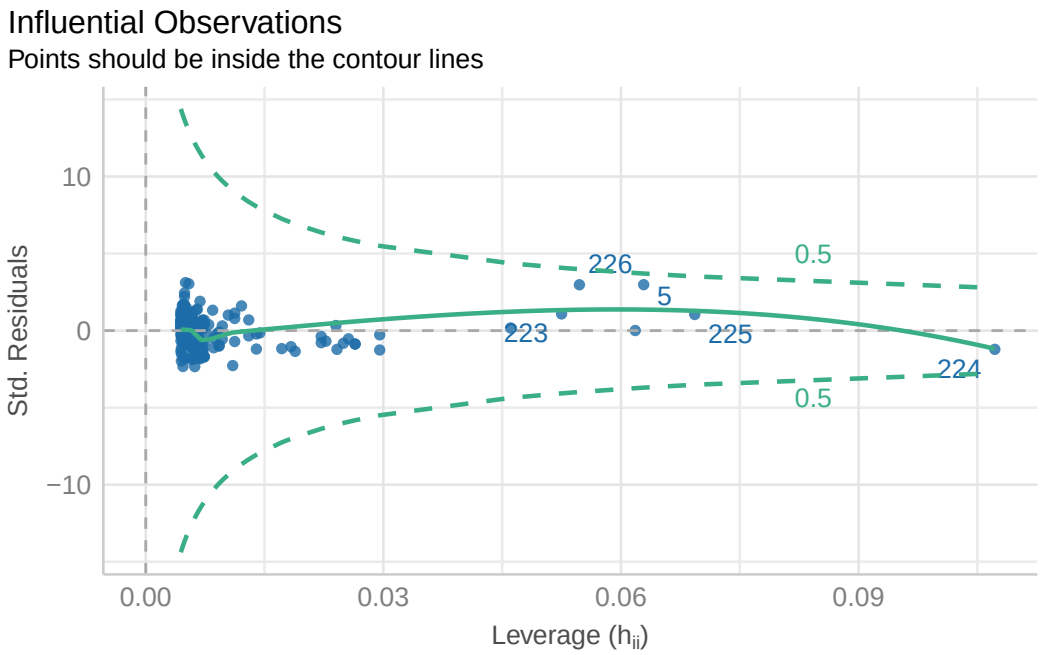
- The scale-location plot is a plot of the square root of the standardized/studentized residuals versus the fitted values.
- The residuals versus leverage plot. Leverage is essentially a measure of how much *potential* an observation has for causing a significant change in the line. If an outlier has relatively high leverage, it may be having a large enough impact on the fitted line, that the line is less accurate because of the outlier.
- `autoplot()` automatically labels which observations you may want to consider investigating by tagging observations with which row, numerically, it is in the data.

There is also the `performance` and `see` R packages that can be help in checking models.

```
library(performance)
library(see)

# Test
check_outliers(beers.lm, method = "cook")
```

```
OK: No outliers detected.
- Based on the following method and threshold: cook (0.7).
- For variable: (Whole model)
```

```
# plot
plot(check_outliers(beers.lm))
```



6.7 Getting Outliers from the Data.

`data[rows,columns]` * Specify the number(s) for which row(s) you want. Leave blank if you want to see all rows. * Specify the number(s) which column(s) you want. Leave blank if you want to see all columns.

Remember, each row in the dataset is a beer. We want to specify just the rows so we can see all the information about each beer we are checking.

```
# Store the indicated outlier numbers as a vector.

outlierRows <- c(198, 218, 226, 5, 224)

beer[outlierRows, ]
```

```
    Calories                          Beer   ABV
198      195         Sam Adams Cream Stout  4.69
218      225           Sierra Nevada Stout  5.80
```

```
226      330         Sierra Nevada Bigfoot  9.60
5         70                       O'Doul's  0.40
224      313 Flying Dog Double Dog\xe5\xca 11.50
```

Maybe there is a pattern here. Maybe these observations should be omitted? If we omit observations, we reduce the **generalizability** of the model. **Generalizability** is the ability for a model to apply to a greater population and future predictions.

6.7.1 New Model Without O'Doul's

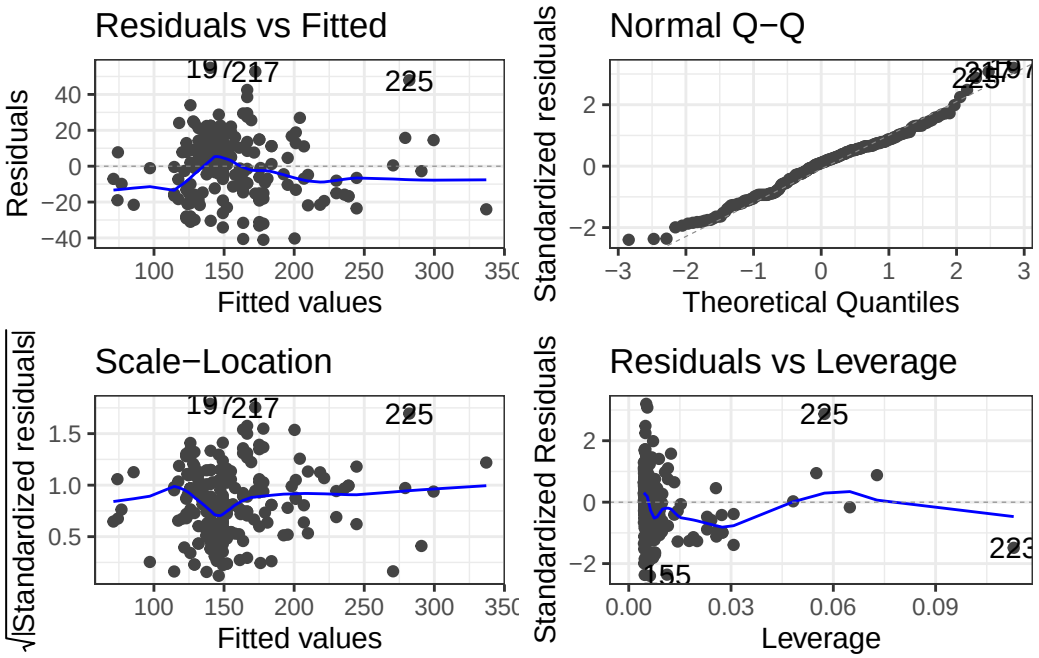
Remove outlier(s):

- Make a new model.
- New residuals check.
- Compare to previous model.

```
beer2 <- filter(beer, Beer != "O'Doul's")

beer2.lm <- lm(Calories ~ ABV, beer2)

summary(beer2.lm)
```

```
Call:
lm(formula = Calories ~ ABV, data = beer2)

Residuals:
    Min      1Q  Median      3Q     Max
-41.046 -14.277   1.753  11.081  54.824

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   4.5978     4.7950   0.959    0.339
ABV          28.9080     0.8966  32.241   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 17.17 on 223 degrees of freedom
Multiple R-squared:  0.8234,    Adjusted R-squared:  0.8226
F-statistic:  1039 on 1 and 223 DF,  p-value: < 2.2e-16
```

```
autoplot(beer2.lm)
```



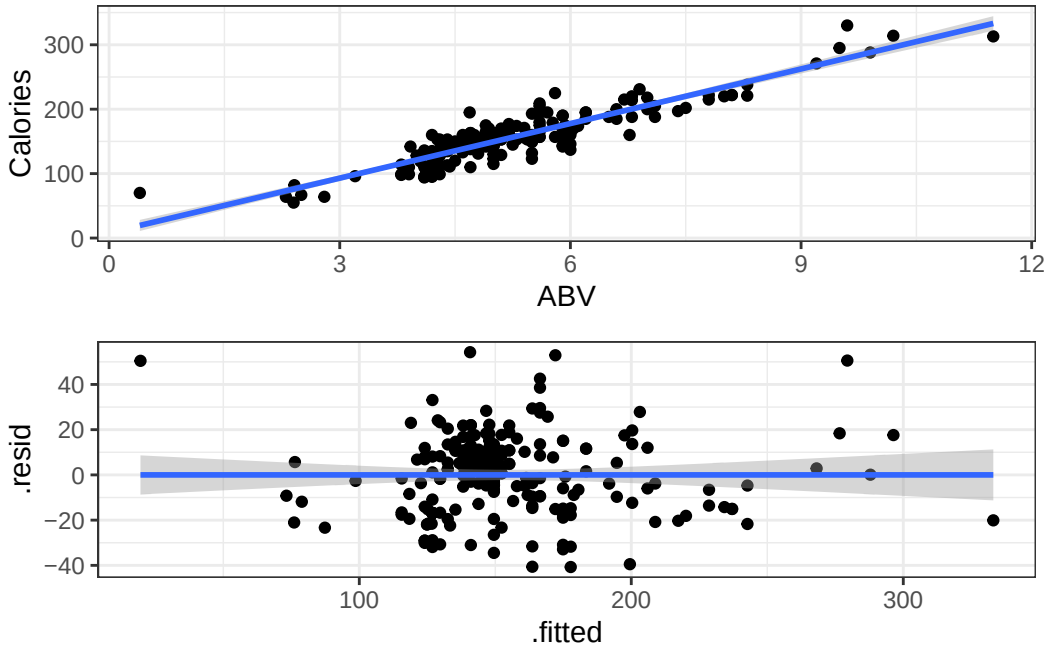
6.8 Specifics of Residual Plots in Simple Linear Regression

One thing to note is that residual plots in simple linear regression are somewhat redundant.

Left: ABV versus Calories

Right: fitted versus residuals

Can you spot the difference? (Besides scale.)



6.8.1 Fitted versus Observed

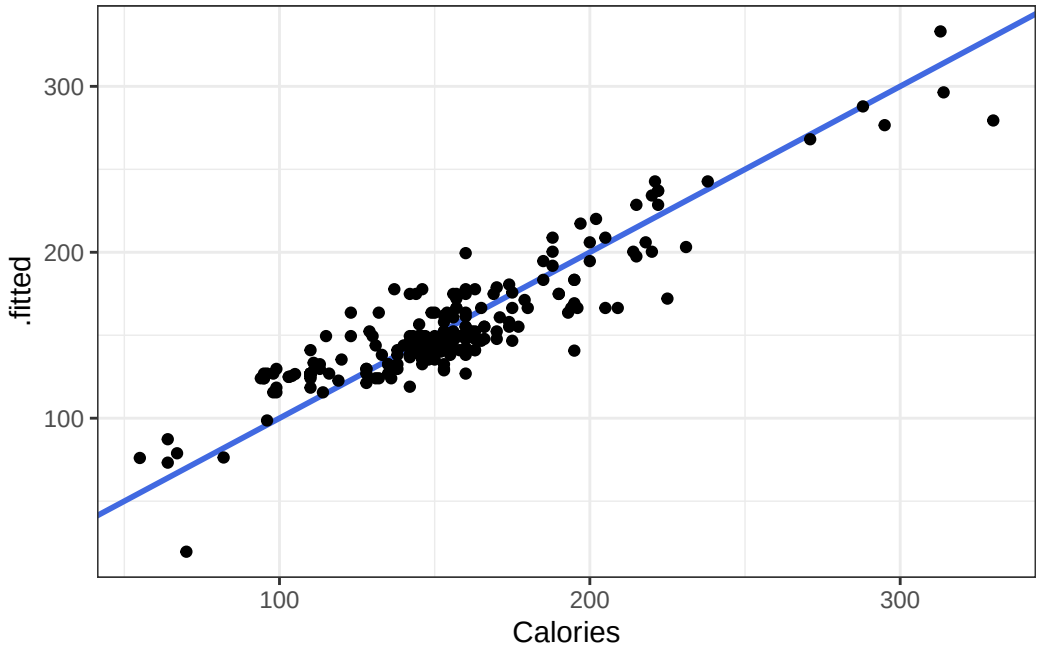
Another alternative plot is that of actual versus predicted.

- Actual y values on one axis.
- Predicted $\hat{y}$ values on the other axis.

You are looking for a 45 degree line with constant dispersion around the line throughout.

Here it is for the beer data.

```
ggplot(beers.lm, aes(x = Calories, y = .fitted)) +
  geom_abline(slope = 1,
              color = "royalblue", size = 1) +
  geom_point()
```



6.8.2 General Model Checks

The R `see` package can also be used to create a general plots for model assumptions

```
check_model(beers.lm)
```

7 Transformations

Here are some code chunks that setup this chapter.

```
# Here are the libraries I used
library(tidyverse) # standard
library(knitr) # need for a couple things to make knitted document to look nice
library(readr) # need to read in data
library(ggpubr) # allows for stat_cor in ggplots
library(ggfortify) # Needed for autoplot to work on lm()
library(gridExtra) # allows me to organize the graphs in a grid
```

```
# This changes the default theme of ggplot
old.theme <- theme_get()
theme_set(theme_bw())
```

Again our model is

$$y = \beta_0 + \beta_1 x + \epsilon$$

where $\epsilon \sim N(0, \sigma)$.

$$x \to g(x)$$

$$h(y) = \beta_0 + \beta_1 g(x) + \epsilon$$

$$y \to h(y)$$

7.1 Not all relations are linear

You'll have to forgive me for a non-biostatistics example, but it exemplifies what I want to discuss in a simple and easily understandable way (I think).

We're going to take a look at some cars.

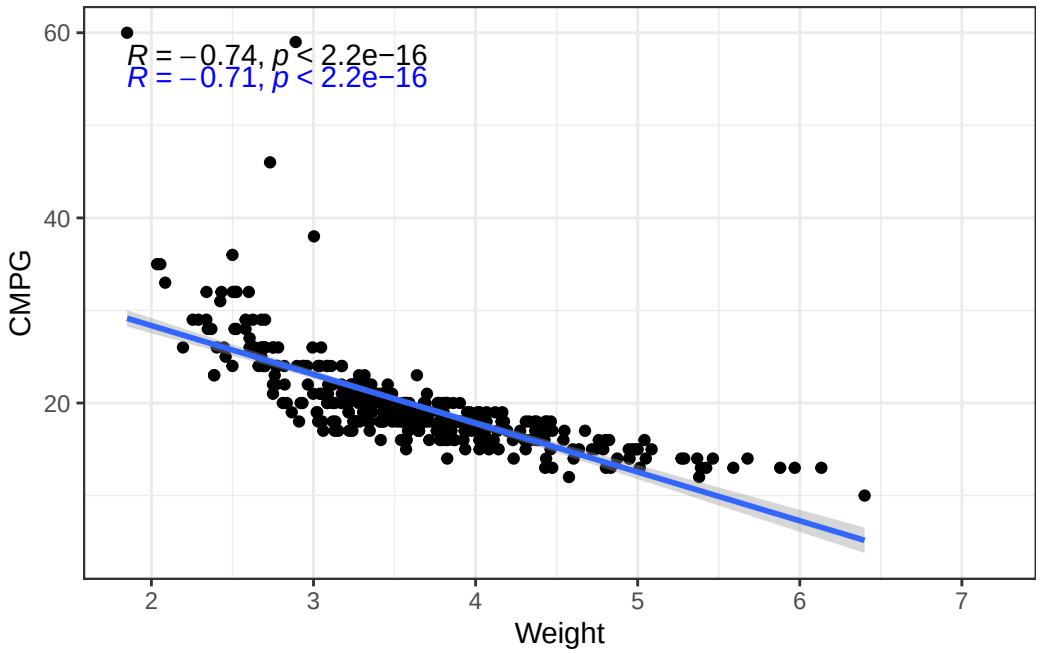
```
cars <- read_csv(here::here("datasets",
                            'cars04.csv'))
```

```
Rows: 428 Columns: 14
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (3): Name, Type, Drive
dbl (11): MSRP, Dealer, Engine, Cyl, HP, CMPG, HMPG, Weight, WheelBase, Leng...

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Several variables to examine, but we'll just parse it down to looking at the relation between the Weight of vehicles and their fuel efficiency in terms of Miles Per Gallon (MPG). `CMPG` is the typical MPG in a city environment, and HMPG is the tyical MPG on highways/interstates/non-stop-and-go driving.

7.1.1 Correlations

```
# I am using arguments in the second stat_cor to change the location of the text
# This is so that the two correlations don't overlap.
ggplot(cars, aes(x = Weight, y = CMPG)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y~x) +
  stat_cor(method = "pearson") +
  stat_cor(method = "kendall", label.y = 55, color = "blue")
```

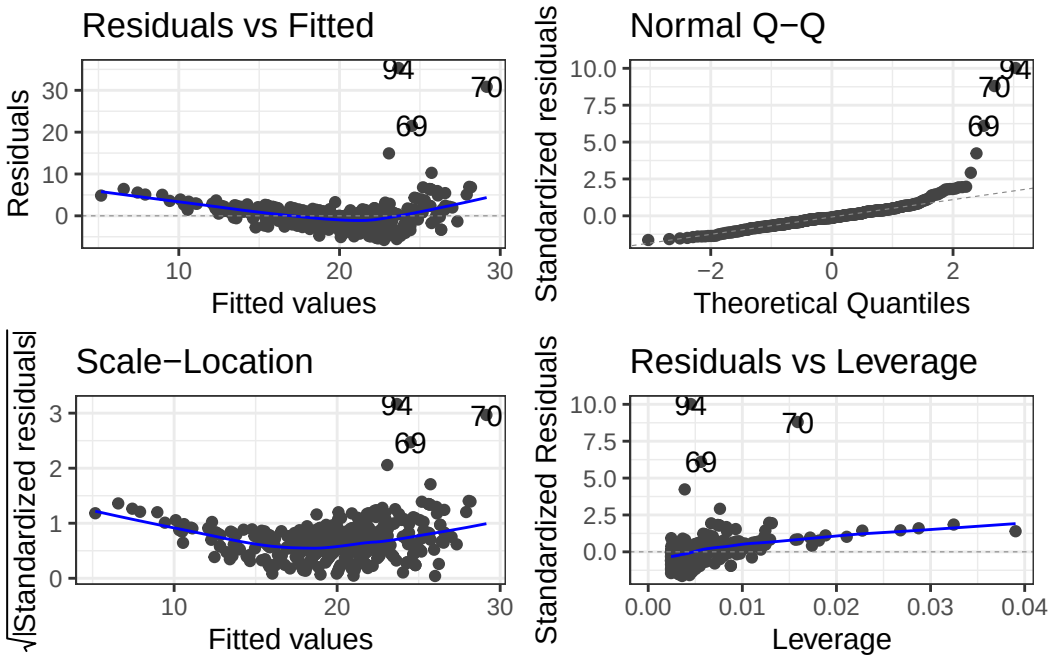


The relation is fairly strong, but does not appear linear. There seems to be a curve to it. The linear model plotted is biased. This means that at some points we expect values to fall below the line more often than not. And other places, we expect the values to fall above the line.

7.1.2 Residuals

This would be reflected residual diagnostics.

```
carsLm <- lm(CMPG ~ Weight, cars)
autoplot(carsLm)
```

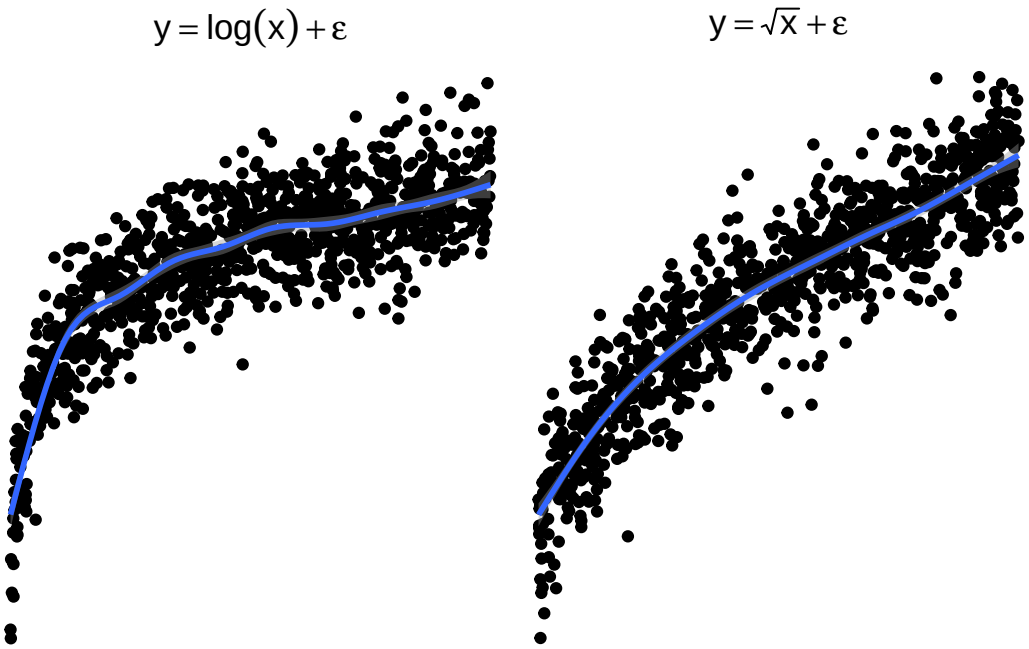


Typically we want to force the data into a linear pattern.

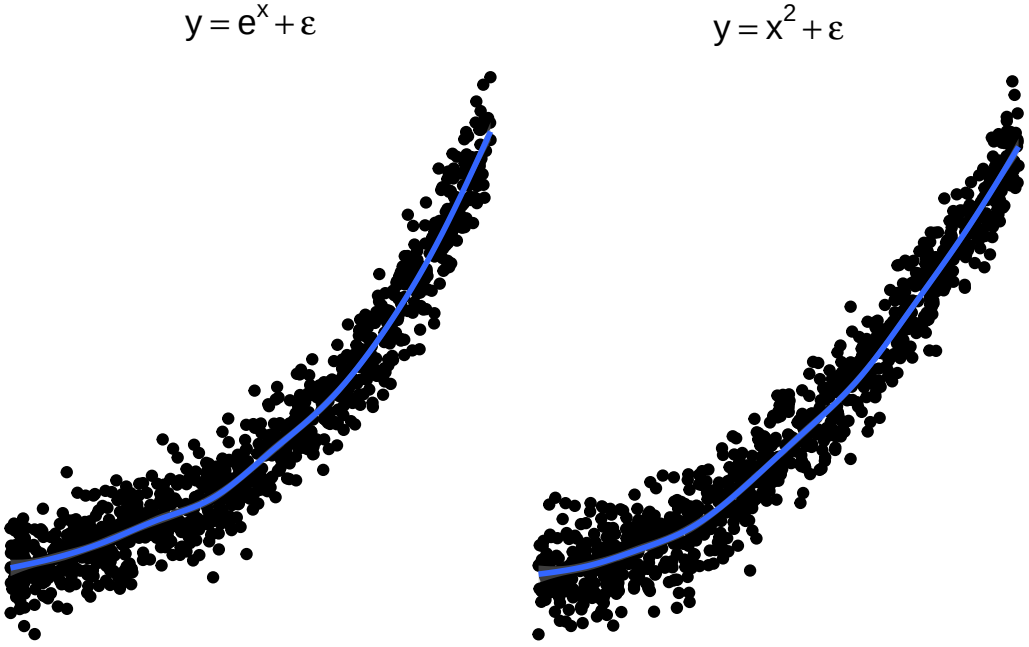
7.1.3 Transformations for Non-linear Relationships

The following plots show several non-linear relationships. With each scatterplot there is the correct transformation for the x variable.

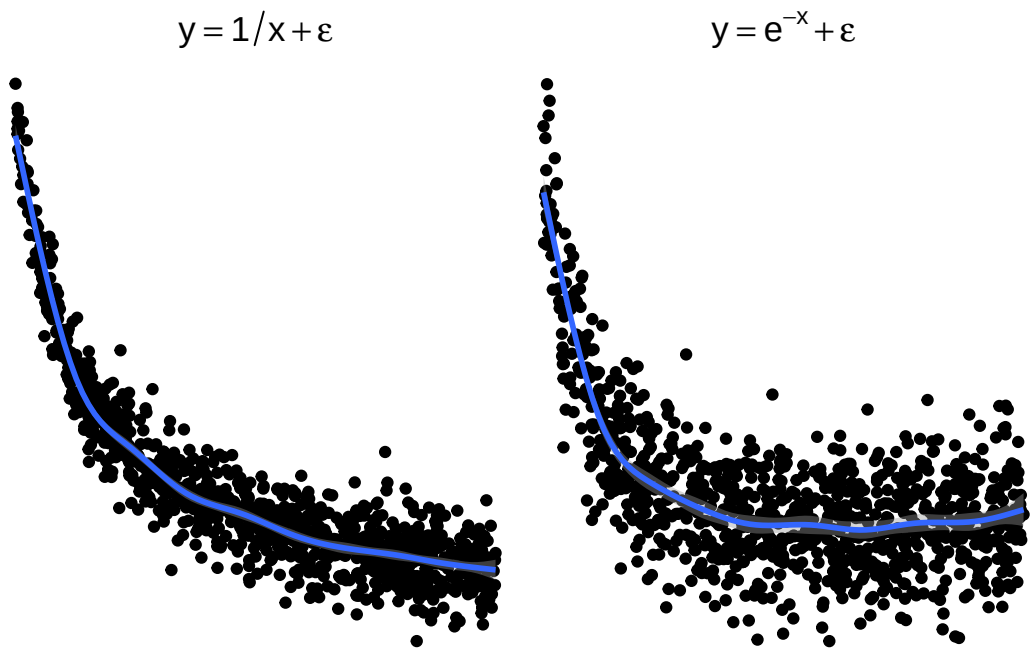
7.1.3.1 $\log(x)$ and $\sqrt{x}$



7.1.3.2 x^2 or e^x (sometimes e^x is denoted by $\exp(x)$)



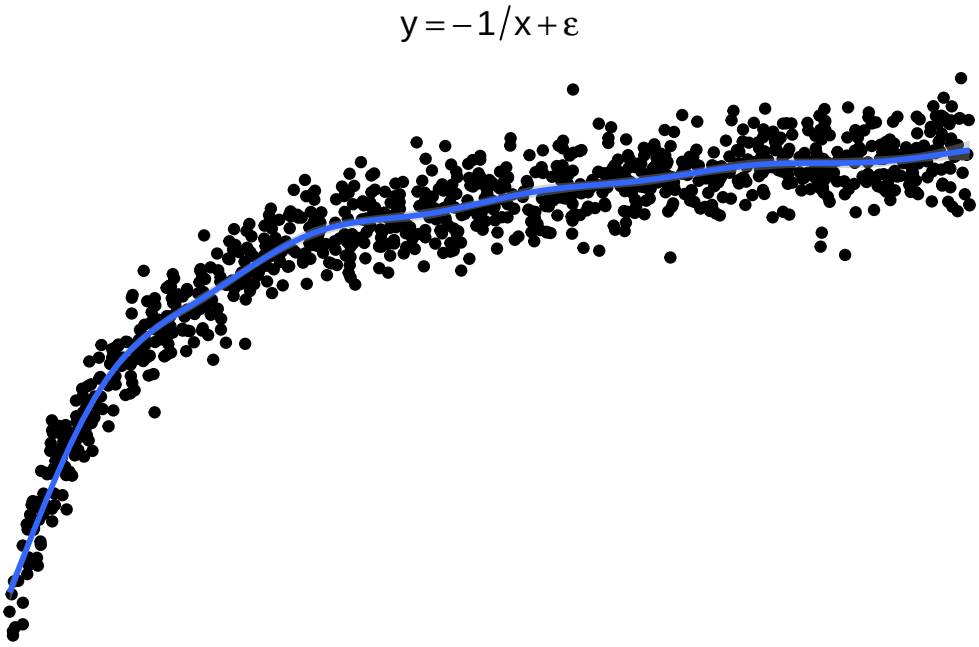
7.1.3.3 $1/x$ or e^{-x} (or $\exp(-x)$)



These are just guidelines for which transformations *may* help. Sometimes the transformations the other transformations may be appropriate because the relationship is flipped by a negative sign.

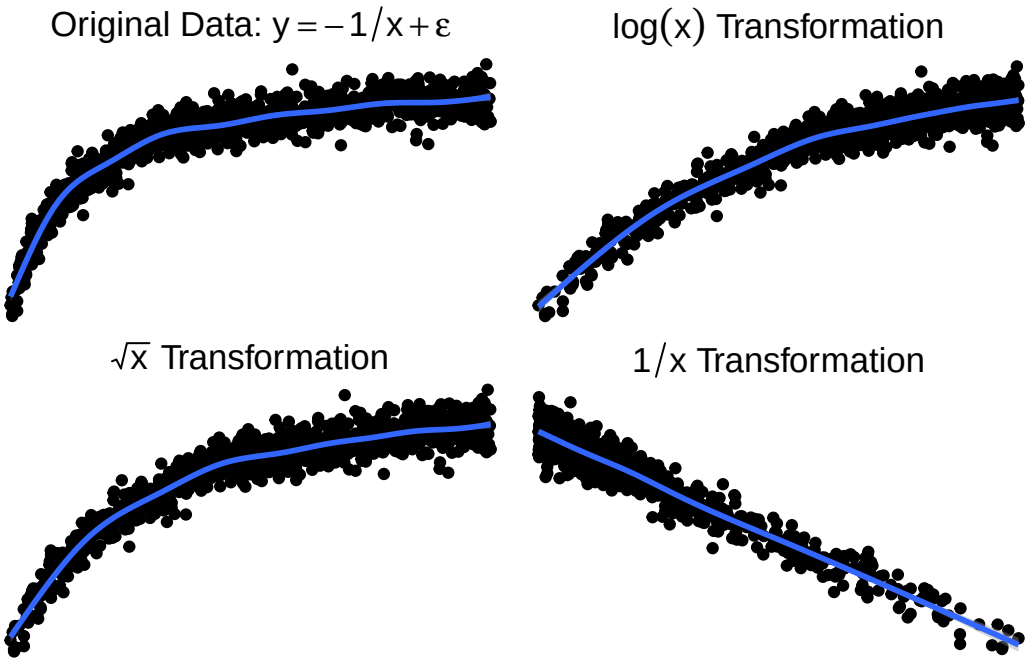
7.1.3.4 Sometimes things look one way and are another

Here is a graph of the model $y = -\frac{1}{x} + \epsilon$



7.1.3.5 Trying $\log(x)$ or $\sqrt{x}$

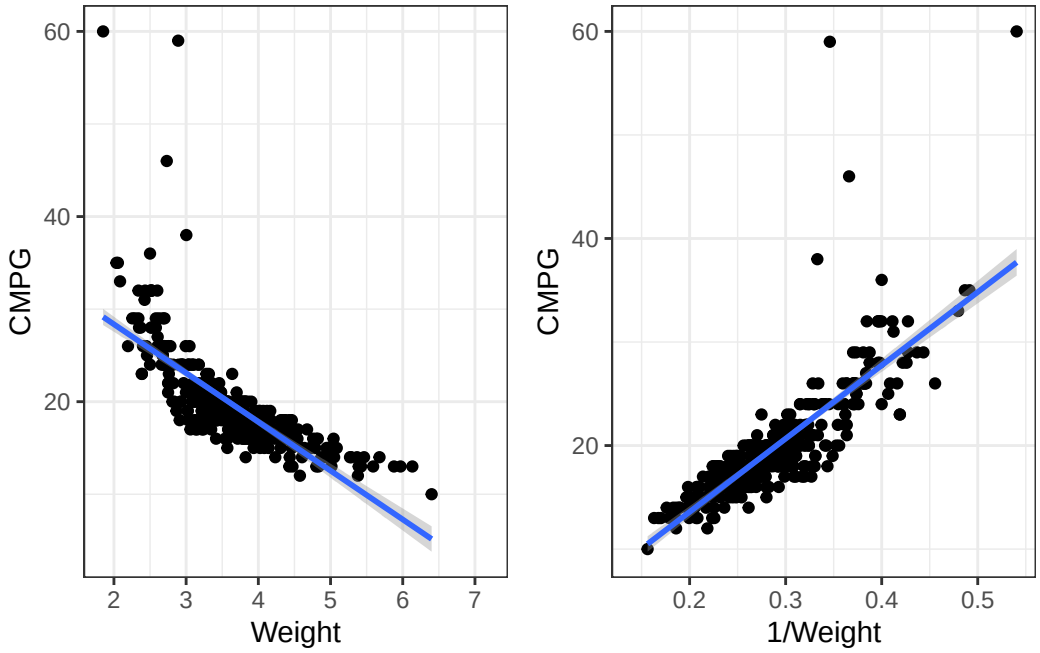
Here are side by side graphs of the original data and transforming the x variable using $\log(x)$, $\sqrt{x}$, and $1/x$.



It is important that you try several transformations before giving up on a linear regression model.

7.1.4 Applying a transformation to the cars dataset

```
p1 <- ggplot(cars, aes(x = Weight, y = CMPG)) +
  geom_point() +
  geom_smooth(method = "lm")

p2 <- ggplot(cars, aes(x = 1/Weight, y = CMPG)) +
  geom_point() +
  geom_smooth(method = "lm")

grid.arrange(p1,p2, nrow = 1)
```



7.1.4.1 Transformations in `lm()`

When performing transformation in linear models, those can be done *within* the `lm()` function. Instead of `lm(y ~ x, data)`, you can put `lm(y ~ I(g(x)), data)`.

- `g(x)` is whatever transformation on the x variable you are trying.
- `I()` is an `R` syntax thing that is needed sometimes so that the function `g(x)` is interpreted correctly.
- Though `I()` is not *always* necessary, it is best practice to use it every time you re performing a transformation.

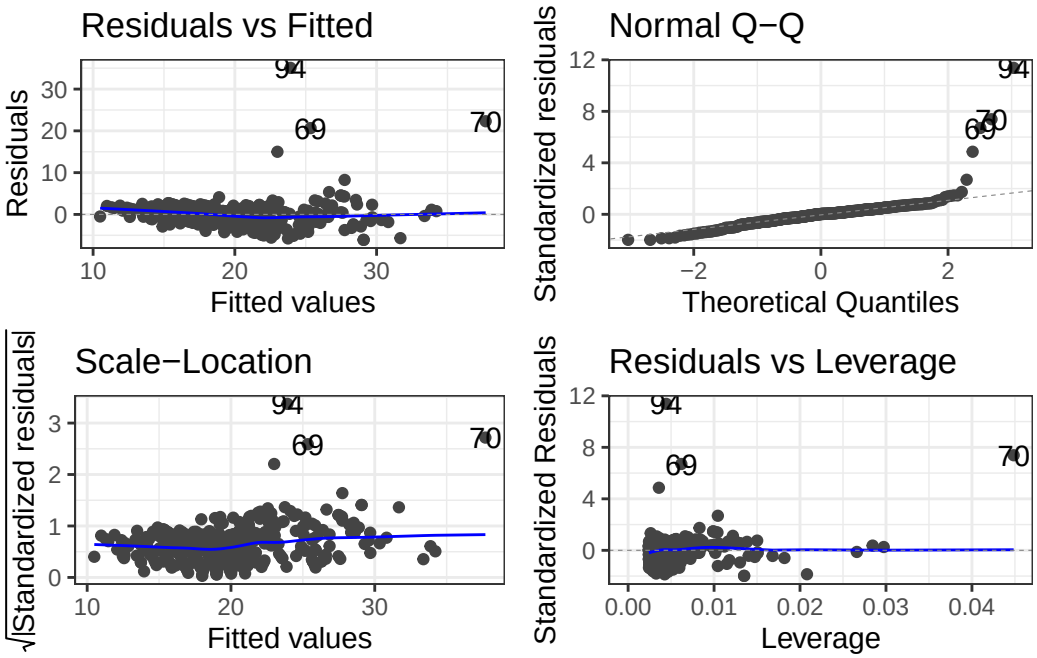
Our function is `1/Weight`, so we would write for the formula, `y ~ I(1/Weight)`. In this case, is *is* necessary to use `I()`. Try summarising a model where the formula is

```
carsLm2 <- lm(CMPG ~ I(1/Weight), cars)
summary(carsLm2)
```

```
Call:
lm(formula = CMPG ~ I(1/Weight), data = cars)

Residuals:
   Min     1Q Median     3Q    Max
-6.088 -1.302  0.069  1.040 35.073

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  -0.5651     0.7629  -0.741    0.459
I(1/Weight)  70.7812     2.5606  27.642   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.09 on 410 degrees of freedom
  (16 observations deleted due to missingness)
Multiple R-squared:  0.6508,    Adjusted R-squared:  0.6499
F-statistic: 764.1 on 1 and 410 DF,  p-value: < 2.2e-16
```

7.1.4.2 Residuals

```
autoplot(carsLm2)
```



7.2 "Stabilizing" Variability

Sometimes the standard deviation of the residuals is constant on a *relative* scale.

$$CV = \frac{\sigma_\epsilon}{|\mu_{y|x}|}$$

The log is typical the transformation used in this situation.

$$\log(y) = \beta_0 + \beta_1 x + \epsilon \qquad \Longleftrightarrow \qquad y = e^{\beta_0 + \beta_1 x + \epsilon}$$

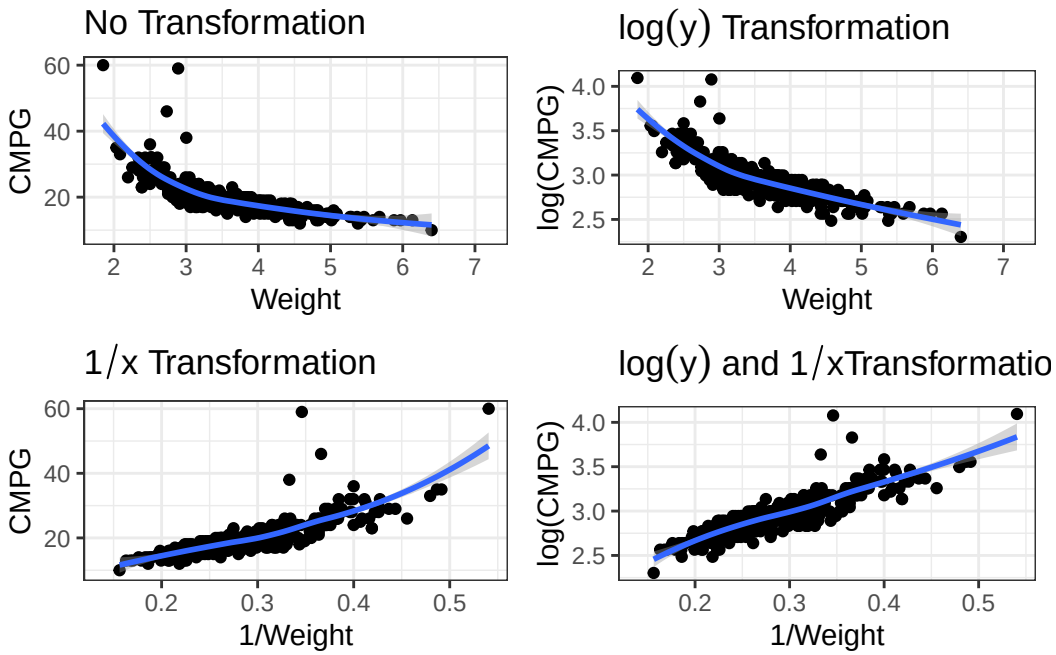
Other possibilities would be taking the square root (or higher power root) of the y variable, but most often it is log that works best if a y transformation is viable.

7.2.1 Log of cars data

There are many potential models. We have to account for non-linearity and the variability issue.

```
p1 <- ggplot(cars, aes(x = Weight, y = CMPG)) +
  geom_point() +
  geom_smooth() +
  labs(title = expression(paste("No Transformation"))) #expression() lets me show math

p2 <- ggplot(cars, aes(x = Weight, y = log(CMPG))) +
  geom_point() +
  geom_smooth() +
  labs(title = expression(paste(log(y), " Transformation"))) #expression() lets me show math

p3 <- ggplot(cars, aes(x = 1/Weight, y = CMPG)) +
  geom_point() +
  geom_smooth() +
  labs(title = expression(paste(1/x, " Transformation"))) #expression() lets me show math

p4 <- ggplot(cars, aes(x = 1/Weight, y = log(CMPG))) +
  geom_point() +
  geom_smooth() +
  labs(title = expression(paste(log(y), " and ", 1/x, "Transformations"))) #expression() lets
```

```
grid.arrange(p1,p2,p3,p4)
```


No Transformation
log(y) Transformation
1/x Transformation
log(y) and 1/xTransformatio
CMPG
log(CMPG)
Weight
1/Weight
60
40
20
4.0
3.5
3.0
2.5
2
3
4
5
6
7
0.2
0.3
0.4
0.5

7.2.1.1 Linear Model

```
carsLm3 <- lm(log(CMPG) ~ I(1/(Weight)), data = cars)

summary(carsLm3)
```

```
Call:
lm(formula = log(CMPG) ~ I(1/(Weight)), data = cars)

Residuals:
     Min       1Q   Median       3Q      Max
-0.25328 -0.05547  0.00506  0.05825  0.92912

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    2.03375    0.02710   75.06   <2e-16 ***
I(1/(Weight))  3.22138    0.09094   35.42   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1098 on 410 degrees of freedom
  (16 observations deleted due to missingness)
Multiple R-squared:  0.7537,    Adjusted R-squared:  0.7531
F-statistic:  1255 on 1 and 410 DF,  p-value: < 2.2e-16
```

```
carsLm3 <- lm(log(CMPG) ~ I(1/(Weight)), data = cars)

summary(carsLm3)
```

```
Call:
lm(formula = log(CMPG) ~ I(1/(Weight)), data = cars)

Residuals:
     Min       1Q   Median       3Q      Max
-0.25328 -0.05547  0.00506  0.05825  0.92912

Coefficients:
```

```
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    2.03375    0.02710   75.06   <2e-16 ***
I(1/(Weight))  3.22138    0.09094   35.42   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1098 on 410 degrees of freedom
  (16 observations deleted due to missingness)
Multiple R-squared:  0.7537,    Adjusted R-squared:  0.7531
F-statistic:  1255 on 1 and 410 DF,  p-value: < 2.2e-16
```

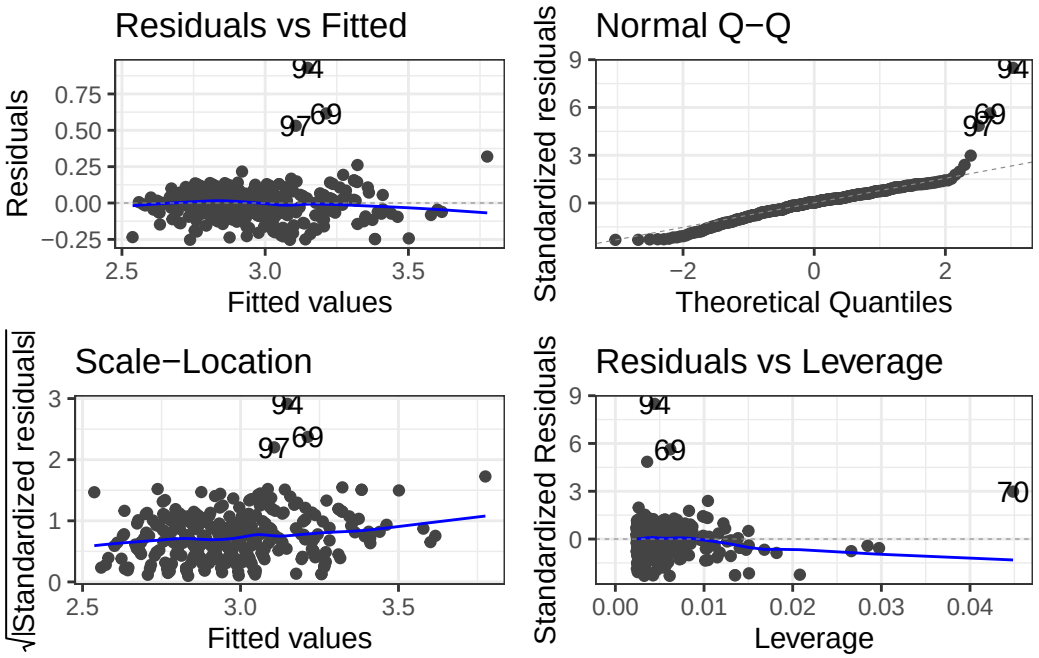
So

$$\widehat{\log(CMPG)} = 2.03 + 3.22\frac{1}{Weight}$$

is our estimated regression line.

7.2.1.2 Residuals

```
autoplot(carsLm3)
```



7.2.1.3 Outliers

```
outliers <- c(94, 69, 97, 70)

cars[outliers, ]
```

```
# A tibble: 4 x 14
  Name  Type  Drive  MSRP Dealer Engine   Cyl    HP  CMPG  HMPG Weight WheelBase
  <chr> <chr> <chr> <dbl>  <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl>  <dbl>     <dbl>
1 Toyo~ Car   FWD   20510  18926    1.5     4   110    59    51   2.89       106
2 Hond~ Car   FWD   20140  18451    1.4     4    93    46    51   2.73       103
3 Volk~ Car   FWD   21055  19638    1.9     4   100    38    46   3.00        99
4 Hond~ Car   FWD   19110  17911    2       3    73    60    66   1.85        95
# i 2 more variables: Length <dbl>, Width <dbl>
```

```
filter(cars, Weight < 2.1)
```

```
# A tibble: 4 x 14
  Name  Type  Drive  MSRP Dealer Engine   Cyl    HP  CMPG  HMPG Weight WheelBase
  <chr> <chr> <chr> <dbl>  <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl>  <dbl>     <dbl>
1 Toyo~ Car   FWD   10760  10144    1.5     4   108    35    43   2.04        93
2 Toyo~ Car   FWD   11560  10896    1.5     4   108    33    39   2.08        93
3 Toyo~ Car   FWD   11290  10642    1.5     4   108    35    43   2.06        93
4 Hond~ Car   FWD   19110  17911    2       3    73    60    66   1.85        95
# i 2 more variables: Length <dbl>, Width <dbl>
```

```
removeOutliers <- c(94, 69, 70)

cars2 <- cars[-removeOutliers, ]
```

7.2.1.4 Removing outliers and new model

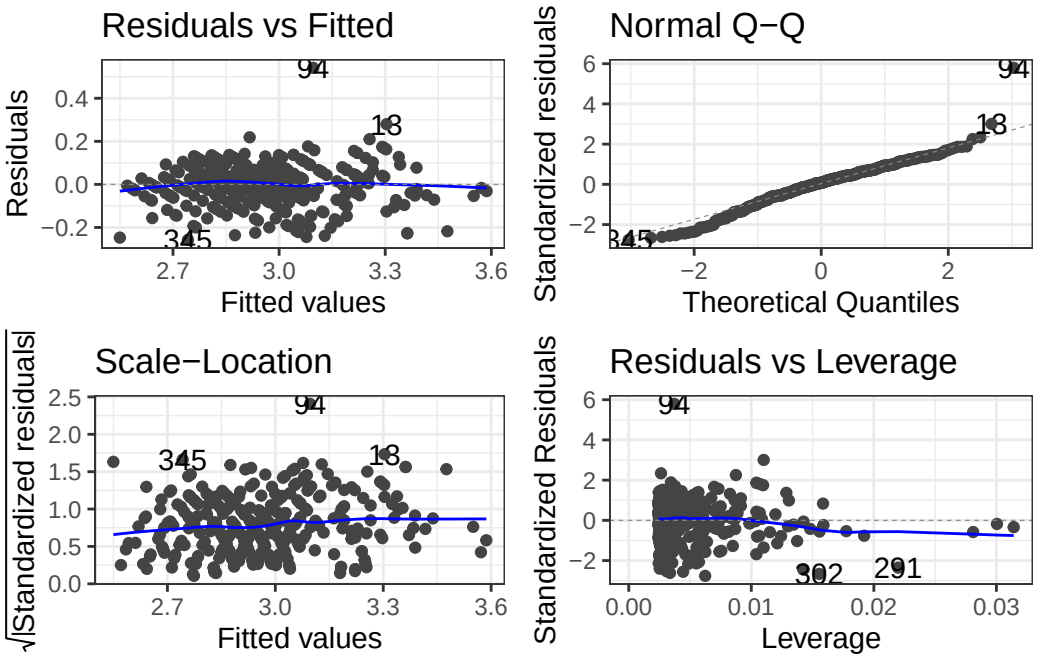
```
carsLm4 <- lm(log(CMPG) ~ I(1/Weight), cars2)

summary(carsLm4)
```

```
Call:
lm(formula = log(CMPG) ~ I(1/Weight), data = cars2)

Residuals:
     Min       1Q   Median       3Q      Max
-0.25752 -0.05228  0.00710  0.05943  0.54103

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.06635    0.02359   87.60   <2e-16 ***
I(1/Weight)  3.09371    0.07948   38.92   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.09357 on 407 degrees of freedom
  (16 observations deleted due to missingness)
Multiple R-squared:  0.7882,    Adjusted R-squared:  0.7877
F-statistic:  1515 on 1 and 407 DF,  p-value: < 2.2e-16
```

Compare to model with outliers.

7.2.1.5 Checking residuals AGAIN

```
autoplot(carsLm4)
```



7.3 You've got a linear model, now what

$$\log(\widehat{CMPG}) = 2.03 + 3.22\frac{1}{Weight} \qquad \Longleftrightarrow \qquad \widehat{CMPG} = e^{2.03+3.22\frac{1}{Weight}}$$

7.3.1 Interpreting you coefficients

We've got this model. How would we explain what it says?

$$\log(\widehat{CMPG}) = 2.03 + 3.22\frac{1}{Weight}$$

This means we are looking at how $1/Weight$ affects the relative change of $CMPG$. What does that mean?

Say you were to look at the average weight of cars that weighed d thousand pounds more. The change in log of CMPG is

$$\Delta \log(CMPG) = \frac{3.22}{Weight + d} - \frac{3.22}{Weight} = \frac{-3.22d}{Weight^2 + d \cdot Weight}$$

Using math "magic".

$$CMPG_d = CMPG_0 \cdot e^{\frac{-3.22d}{Weight^2 + d \cdot Weight}}$$

In terms of viewing the model as $\widehat{CMPG} = e^{2.03+3.22\frac{1}{Weight}}$ the change in $CMPG$ would be

$$\Delta\widehat{CMPG} = \left(e^{2.03+3.22\frac{1}{Weight}}\right) - \left(e^{2.03+3.22\frac{1}{Weight+d}}\right)$$

I don't know about this form either.

7.3.2 Predictions from transformations

You create a `data.frame` that contains the values of the predictor variable that you want predictions for, and then plug them into the `predict()` function.

```
newdata <- data.frame(Weight = c(1.5, 2, 2.5, 3, 3.5, 4))

predictions <- predict(carsLm4, newdata)
```

Now let's look at those predictions.

```
predictions
```

```
       1        2        3        4        5        6
4.128824 3.613205 3.303834 3.097587 2.950267 2.839777
```

There's an issue here. These are the predictions for $\log(CMPG)$, not CMPG. You have to use the reverse function on the predictions. In this case, `log(y)` takes the natural log of y. Therefore, your reverse function is e^y using `exp(y)`.

```
exp(predictions)
```

```
       1        2        3        4        5        6
62.10485 37.08473 27.21679 22.14445 19.11106 17.11196
```

7.3.2.1 Confidence and Interval Intervals

Likewise, if you wanted confidence intervals for the mean, or prediction intervals for future observations, you would specify an `interval = "confidence"` or `interval = "prediction"` argument and a `level` argument specifying the desired confidence level for your intervals. AND you need to apply the reverse function.

```
CIs <- predict(carsLm4, newdata, interval = "confidence", level = 0.99)

PIs <- predict(carsLm4, newdata, interval = "prediction", level = 0.99)
```

Here are the 99% confidence intervals.

```
exp(CIs)
```

```
       fit      lwr      upr
1 62.10485 57.43392 67.15565
2 37.08473 35.46636 38.77696
3 27.21679 26.53386 27.91730
4 22.14445 21.81908 22.47467
5 19.11106 18.88265 19.34223
6 17.11196 16.86310 17.36448
```

And here are the 99% prediction intervals.

```
exp(PIs)
```

```
       fit      lwr      upr
1 62.10485 48.15157 80.10149
2 37.08473 28.99050 47.43889
3 27.21679 21.33490 34.72030
4 22.14445 17.37399 28.22475
5 19.11106 14.99637 24.35473
6 17.11196 13.42575 21.81026
```

Confidence levels when you apply the reverse transformations are only approximate. Actual confidence may be higher or lower. Something about this thing called Jensen's inequality…

8 Introduction to Multiple Regression

Here are some code chunks that setup this chapter

```
# Here are the libraries I used
library(tidyverse) # standard
library(knitr) # need for a couple things to make knitted document to look nice
library(readr) # need to read in data
library(ggpubr) # allows for stat_cor in ggplots
library(ggfortify) # Needed for autoplot to work on lm()
library(gridExtra) # allows me to organize the graphs in a grid
library(car)
```

```
# This changes the default theme of ggplot
old.theme <- theme_get()
theme_set(theme_bw())
```

8.1 SENIC Data

We will now begin the examining data from the Study on the Efficcacy of Nosocomial Infection Control (*SENIC* project). The general objective, and therefore project name, was to examine how effective infection surveillance and control programs were at reducing hopsital acquired (nosocomial) diseases. Data was obtained through the test Applied Linear Statistical Models 5th edition (Neter et al).

```
senic <- read_csv(here::here("datasets",
                             'SENIC.csv'))
```

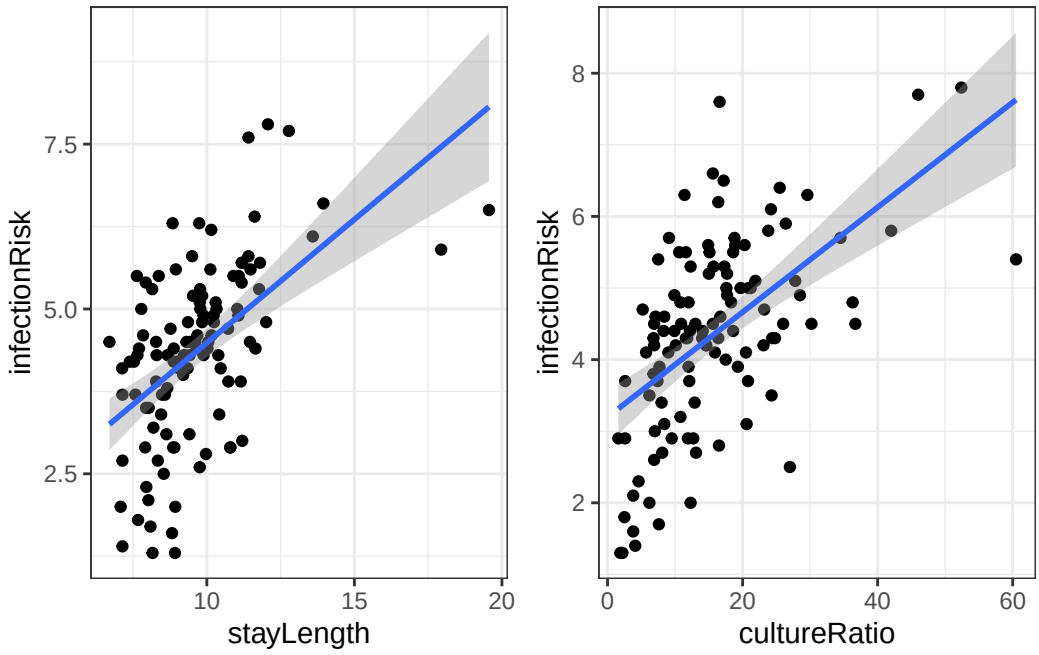
- `stayLength`: Average length of stay of all patients in hospital (in days)
- `age`: Average age of patients (in years)
- `infectionRisk`: Average estimated probability of acquiring infection in hospital (in percent)
- `cultureRatio`: Ratio of number of cultures performed to number of patients without signs or symptoms of hospital-acquired infection, times 100
- `xrayRatio`: Ratio of number of X-rays performed to number of patients , without signs or symptoms of pneumonia, times 100
- `beds`: Average number of beds in hospital during study period
- `school` Med school affiliation (Yes or No)
- `region`: Geographic region (NE, NC, S, W)
- `patients`: Average number of patients in hospital per day during study period
- `nurses`: Average number of full-time equivalent registered and licensed practical nurses during study period (number full-time plus one half the number part time)
- `facilities`: Percent of 35 potential facilities and services that are provided by the hospital

8.2 Infection Risk

There are a lot of potential variables that may relate to `infectionRisk`, but let's just concentrate on `stayLength` and `cultureRatio`.

```
#isc ending indicates the variables used:
#(i)nfectionRisk, (s)tayLength, (c)ultureRatio

senicisc <- select(senic, infectionRisk,
                   stayLength, cultureRatio)
```

8.2.1 Relation of `infectionRisk` with `stayLength` and `cultureRatio`



8.2.2 Model with `stayLength`

Here is the linear regression output for `infectionRisk` with `stayLength`.

```
infStayLm <- lm(infectionRisk ~ stayLength, senicisc)
summary(infStayLm)
```

```
Call:
lm(formula = infectionRisk ~ stayLength, data = senicisc)

Residuals:
    Min      1Q  Median      3Q     Max
-2.7823 -0.7039  0.1281  0.6767  2.5859

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.74430    0.55386   1.344    0.182
stayLength   0.37422    0.05632   6.645 1.18e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.139 on 111 degrees of freedom
Multiple R-squared:  0.2846,    Adjusted R-squared:  0.2781
F-statistic: 44.15 on 1 and 111 DF,  p-value: 1.177e-09
```

Our regression equation for predicting `infectionRisk` is:

$$\widehat{risk} = 0.744 + 0.374 \cdot stayLength$$

8.2.3 Model with `cultureRatio`

Here is the linear regression output for `infectionRisk` with `cultureRatio`.

```
infCuLm <- lm(infectionRisk ~ cultureRatio, senicisc)
summary(infCuLm)
```

```
Call:
lm(formula = infectionRisk ~ cultureRatio, data = senicisc)

Residuals:
    Min      1Q  Median      3Q     Max
-2.6759 -0.7133  0.1593  0.7966  3.1860

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   3.19790    0.19377  16.504  < 2e-16 ***
cultureRatio  0.07326    0.01031   7.106 1.22e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.117 on 111 degrees of freedom
Multiple R-squared:  0.3127,    Adjusted R-squared:  0.3065
F-statistic: 50.49 on 1 and 111 DF,  p-value: 1.218e-10
```

The regression equation is then:

$$\widehat{risk} = 3.198 + 0.073 \cdot cultureRatio$$

8.3 Linear Regression with Two Variables

We can "easily" create a model that accounts for a linear relationship between two variables. If we have one response variable y and two predictor variables x_1 and x_2, then the model is:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \epsilon$$

Where we assume that we have error terms $\epsilon \sim N(0, \sigma_\epsilon)$ which are independent. *An additional assumption is that the two predictor variables, x_1 and x_2 are independent of each other.*

We still have two parts to the model.

- The linear relation/conditional mean $\mu_{y|x} = \beta_0 + \beta_1 x_1 + \beta_2 x_2$
- The error ϵ

8.4 Model for `infectionRisk` using two variables

The way to get the estimated regression equation is the same. We just add another variable into the `lm()` formula.

```
riskLm <- lm(infectionRisk ~ stayLength + cultureRatio,
             data=senicisc)

summary(riskLm)
```

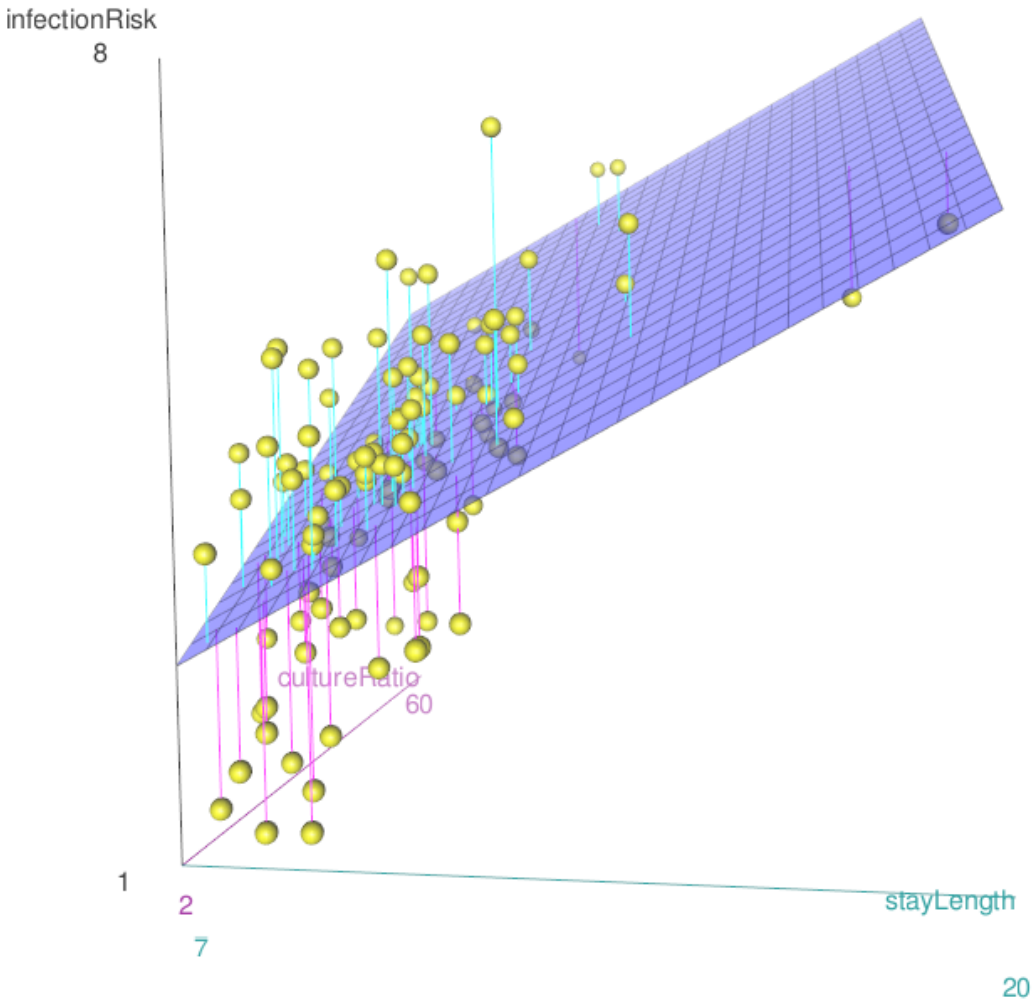
```
Call:
lm(formula = infectionRisk ~ stayLength + cultureRatio, data = senicisc)

Residuals:
    Min      1Q  Median      3Q     Max
-2.1822 -0.7275  0.1040  0.6847  2.7143

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.805491   0.487756   1.651    0.102
stayLength   0.275472   0.052465   5.251 7.46e-07 ***
cultureRatio 0.056451   0.009798   5.761 7.70e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 1.003 on 110 degrees of freedom
Multiple R-squared:  0.4504,    Adjusted R-squared:  0.4404
F-statistic: 45.07 on 2 and 110 DF,  p-value: 5.04e-15
```

So our model for estimated risk is

$$\widehat{risk} = 0.805 + 0.275 \cdot stayLength + 0.056 \cdot cultureRatio$$

8.4.1 Graphing the relationship



8.4.2 Interpreting the Coefficients

The interpretation, so now we have to figure out to interpret three different estimated coefficients: $\hat{\beta}_0, \hat{\beta}_1, \hat{\beta}_2$.

- $\hat{\beta}_0$ is the estimated y-intercept. The idea behind the intercept coefficient is similar, except for now it is $\hat{\mu}_{y|x}hat$, the estimated mean value of the y variable, when *both* x_1 and x_2 are 0.
- $\hat{\beta}_1$ is the amount that $\hat{\mu}_{y|x}$ increases when x_1 increases by 1 unit AND x_2 is constant.
- $\hat{\beta}_2$ is the amount that $\hat{\mu}_{y|x}$ increases when x_2 increases by 1 unit AND x_1 is constant.

So for our estimated regression model:

$$\widehat{risk} = 0.805 + 0.275 \cdot stayLength + 0.056 \cdot cultureRatio$$

- We estimate that the mean infection risk of hospitals is 0.805% among hospitals with an average length of stay if 0 days and average culture ratio of 0. (Does this make sense?)
- We estimate that the mean infection risk of hospitals increases by 0.275% among hospitals with average length of stay 1 day longer, and average culture ratio saying constant.
- We estimate that the mean infection risk of hospitals increases by 0.056% among hospitals where the average culture ratio rate is 1 unit higher, and average length of stay remains constant.

8.5 Inference on the regression coefficients

Just like before, there are several columns for each coefficient. The columns are for the following hypothesis test.

$$H_0 : \beta_k = 0$$
$$H_1 : \beta_k \neq 0$$

The test statistic is:

$$t = \frac{\hat{\beta}_k}{SE_{\hat{\beta}_k}}$$

We simple divide our coefficient estimate by its standard error.

The test statistic is assumed to be from a t distribution with degrees of freedom $df = n-(p+1)$. Where p is the total number of predictor variables. And the p-value is the probability of obtaining

When looking at our model summary we interpret the p-values similarly.

```
summary(riskLm)
```

```
Call:
lm(formula = infectionRisk ~ stayLength + cultureRatio, data = senicisc)

Residuals:
    Min      1Q  Median      3Q     Max
-2.1822 -0.7275  0.1040  0.6847  2.7143

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.805491   0.487756   1.651    0.102
stayLength   0.275472   0.052465   5.251 7.46e-07 ***
cultureRatio 0.056451   0.009798   5.761 7.70e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.003 on 110 degrees of freedom
Multiple R-squared:  0.4504,    Adjusted R-squared:  0.4404
F-statistic: 45.07 on 2 and 110 DF,  p-value: 5.04e-15
```

- There is weak evidence that the true linear model has a non-zero intercept.
- There is extremely strong evidence that there is a linear component in the relationship between `stayLength` and `infectionRisk` ($p < 0.0001$) when `cultureRatio` is included in the model.
- There is extremely strong evidence that there is a linear component in the relationship between `cultureRatio` and `infectionRisk` ($p < 0.0001$) when `stayLength` is included in the model.

The test for the intercept is not important (usually). The tests for the slopes are more important though not terribly so by themselves.

8.5.1 Confidence Intervals for Coefficients

We can likewise get confidence intervals for the coefficients. They take the general form:

$$\hat{\beta}_i \pm t_{\alpha/2,df} \cdot SE_{\hat{\beta}_i}$$

Again the $t_{\alpha/2}$ quantile depends on the value for α of the desired nominal confidence level $1 - \alpha$.

To get the confidence intervals we use the `confint()` function just before.

```
confint(riskLm, level = 0.99)
```

```
                 0.5 %     99.5 %
(Intercept)  -0.4730459 2.08402798
stayLength    0.1379482 0.41299604
cultureRatio  0.0307672 0.08213574
```

The estimated change in the mean of `infectionRisk` is:

- From 0.138 to 0.413 at 99% confidence when increasing `stayLength` by 1 day. (And keeping `cultureRatio` constant.)
- From 0.031 to 0.082 at 99% confidence when increasing `cultureRatio` by 1 (ratios don't really have units). (Keeping `stayLength` constant.)

8.6 Estimating the Mean/Predicting Future Observations

Getting predictions is similar to before. You use the `predict()` function, and you need to specify what values you want predictions for.

You make a `data.frame` with values for each predictor variable.

- You must specify *all* variables that are used as predictors.
- Each variable must have the same number of points.
- If you screw this up, you will get predictions for all of the original points in the data.

Lets say you want to predict/estimate `infectionRisk` for a hospital that have an average length of stay of 5, 10, and 15 days, and a average culture ratio of 5, 15, 25. But which way are we combining those?

```
newdata <- data.frame(stayLength = c(5, 10, 15),
                      cultureRatio = c(5, 15, 25))
predict(riskLm, newdata)
```

```
       1        2        3
2.465109 4.406984 6.348859
```

Or what about?

```
newdata <- data.frame(stayLength = c(5, 10, 15),
                      cultureRatio = c(25, 15, 5))
predict(riskLm, newdata)
```

```
       1        2        3
3.594138 4.406984 5.219830
```

Or...?

```
newdata <- data.frame(stayLength = c(15, 5, 10),
                      cultureRatio = c(15, 25, 5))
predict(riskLm, newdata)
```

```
       1        2        3
5.784345 3.594138 3.842469
```

You have to know which exact combination of predictor variable values you want.

8.6.1 Confidence Intervals for the Mean and Prediction Intervals for Future Observations

This is the same as well.

- Confidence intervals (CIs) are establishing a range of plausible values for the conditional mean: $\mu_{y|x} = \beta_0 + \beta_1 x_1 + \widehat{\beta}_2 x_2$
- Prediction intervals (PIs) establish a range of plausible values for individual observations: $y = \beta_0 + \beta_1 x_1 + \widehat{\beta}_2 x_2 + \epsilon$

Let's make 99% CIs and PIs.

```
newdata <- data.frame(stayLength = c(15, 5, 10),
                      cultureRatio = c(15, 25, 5))
CIs <- predict(riskLm, newdata,
               interval = "confidence", level = 0.99)
PIs <- predict(riskLm, newdata,
               interval = "prediction", level = 0.99)
```

```
CIs
```

```
       fit      lwr      upr
1 5.784345 5.001363 6.567327
2 3.594138 2.803874 4.384403
3 3.842469 3.456304 4.228635
```

```
PIs
```

```
       fit       lwr      upr
1 5.784345 3.0409109 8.527779
2 3.594138 0.8486173 6.339659
3 3.842469 1.1849346 6.500004
```

Interpretations are now relative to the value of both predictors variables.

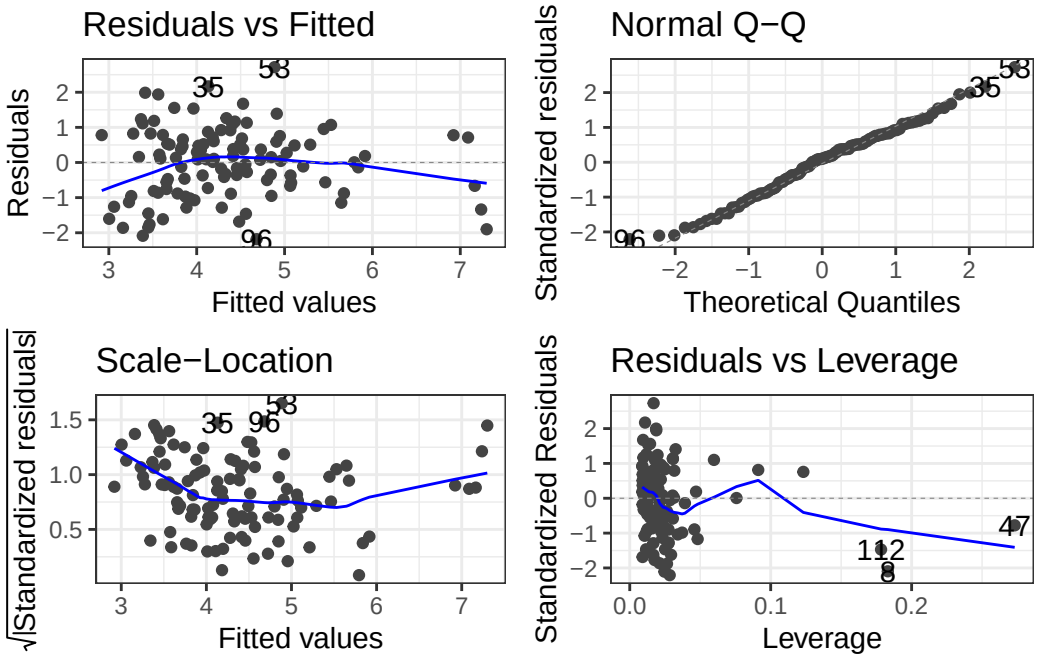
CI Interpretation: There is 99% confidence that the mean infection risk of hospitals with average length of stay of 15 days and culture ratio of 15 is between 5.00% to 6.57%.

PI Interpretation: There is 99% confidence that the infection risk of an *Individual* hospital with average length of stay of 15 days and culture ratio of 15 is between 3.04% to 8.53%.

8.7 Residual Analysis

To assess validity of the model assumptions we can examine the residuals in the same manner as before. Create plots of the residuals using `autoplot()` and assess them in the same way as before.

```
autoplot(riskLm)
```



8.8 Adding more variables!

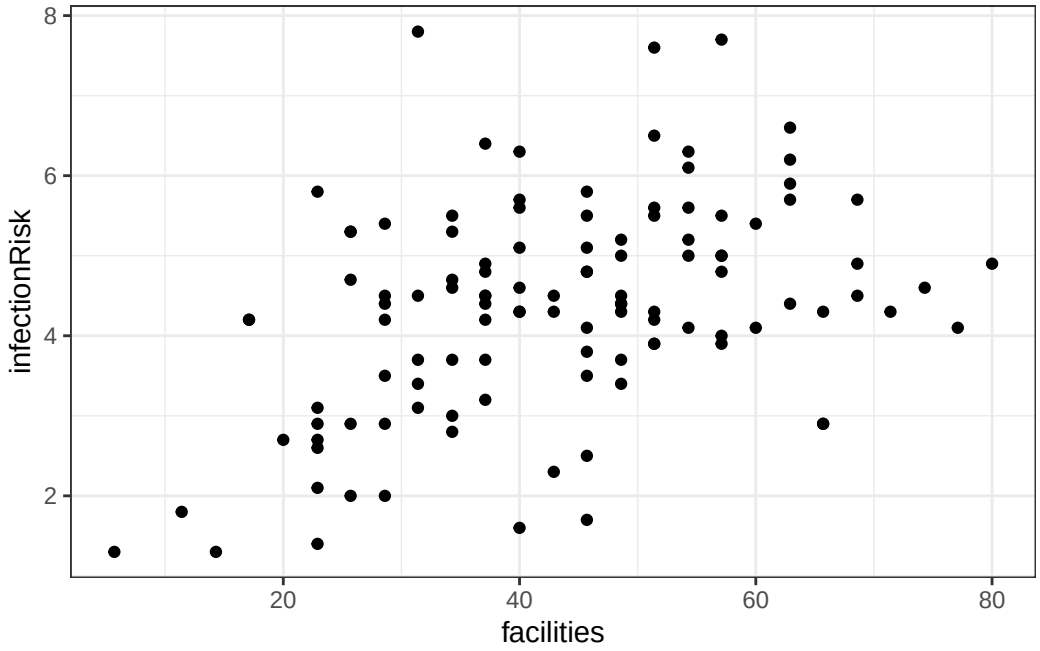
The linear model is not restricted to one or two variables. We can have as many predictor variables as we want! Let p denote the total number of predictor variables we use. The linear model is now:

$$y = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + ... + \hat{\beta}_p + \epsilon = \beta_0 + \sum_{k=1}^{p} \beta_k x_k + \epsilon$$

8.8.1 `facilities` and `infectionRisk`?

Here is a graph between `facilities` and `infectionRisk`.

```
ggplot(senic, aes(x = facilities, y = infectionRisk)) +
  geom_point()
```



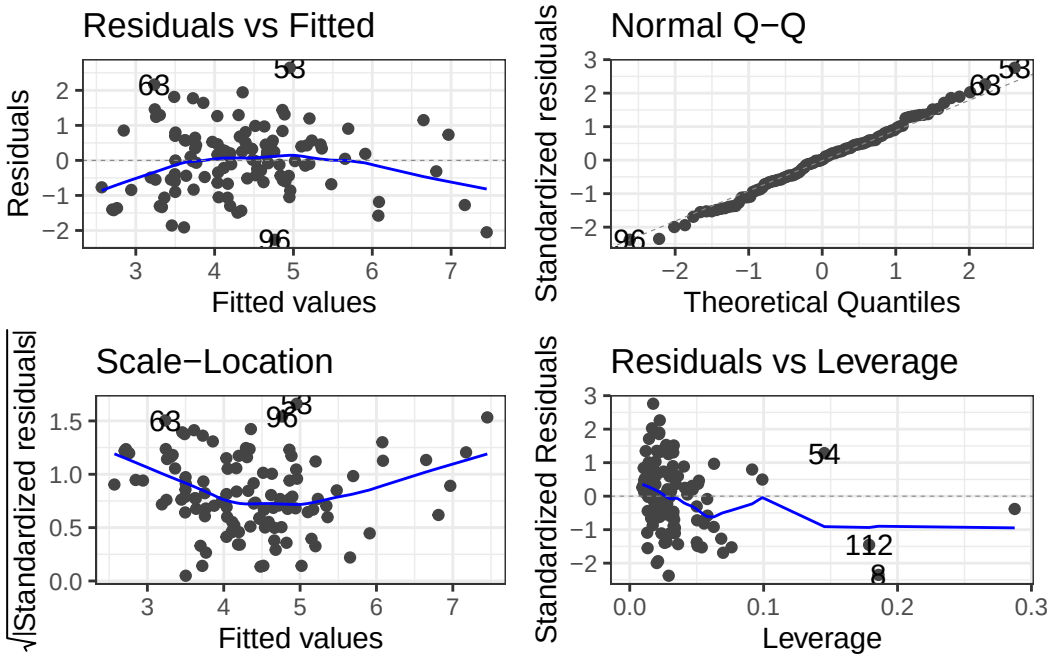
8.8.2 Adding `facilities` to the `infectionRisk` model.

```
riskierLm <- lm(infectionRisk ~ stayLength + cultureRatio + facilities, senic)
summary(riskierLm)
```

```
Call:
lm(formula = infectionRisk ~ stayLength + cultureRatio + facilities,
    data = senic)

Residuals:
     Min       1Q   Median       3Q      Max
-2.26400 -0.59873  0.01723  0.56650  2.64517

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.491332   0.481636   1.020  0.30992
stayLength   0.223907   0.053366   4.196 5.56e-05 ***
cultureRatio 0.054200   0.009479   5.718 9.55e-08 ***
facilities   0.019630   0.006454   3.042  0.00295 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9674 on 109 degrees of freedom
Multiple R-squared:  0.4934,    Adjusted R-squared:  0.4795
F-statistic: 35.39 on 3 and 109 DF,  p-value: 4.769e-16
```

8.8.3 Remember to always check your residuals!

```
autoplot(riskierLm)
```



8.8.4 The Model Analysis of Variance: Global F-Test

Assuming there are p predictor variables

$H_0 : \beta_1 = \beta_2 = \cdots = \beta_p = 0$

H_1 : At least one β_k is not 0.

To test this, we use that same breakdown of model variability as before:

$$SSTO = \sum_{i=1}^{n}(y_i - \bar{y})^2$$

This variability in the variable can be broken up into two pieces:

$$SSR = \sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2 = \sum_{i=1}^{n}(\hat{\mu}_{y|x} - \hat{\mu}_y)^2$$

$$SSE = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

The two sums of squares combined are the total variability $SSTO$.

$$SSTO = SSR + SSE$$

The Regression Degrees of Freedom is now p, the error degrees of freedom is $n - (p + 1)$.

This gives the following mean squares

- Mean Square Regression: $MSR = SSR/p$
- Mean Square Error: $MSE = SSE/[n - (p + 1)]$

The test statistic is $F_t = MSR/MSE$ and we use the $F(p, n-(p+1))$ distribution to compute the p-value.

$$F_t = \frac{SSR \div p}{MSE} = \frac{MSR}{MSE}$$

8.8.5 F-Test for `infectionRisk` model with 3 predictors

This test is available via the model summary in the very last row.

```
summary(riskierLm)
```

```
Call:
lm(formula = infectionRisk ~ stayLength + cultureRatio + facilities,
    data = senic)

Residuals:
     Min       1Q   Median       3Q      Max
-2.26400 -0.59873  0.01723  0.56650  2.64517

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.491332   0.481636   1.020  0.30992
stayLength   0.223907   0.053366   4.196 5.56e-05 ***
cultureRatio 0.054200   0.009479   5.718 9.55e-08 ***
facilities   0.019630   0.006454   3.042  0.00295 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9674 on 109 degrees of freedom
Multiple R-squared:  0.4934,    Adjusted R-squared:  0.4795
F-statistic: 35.39 on 3 and 109 DF,  p-value: 4.769e-16
```

Does this really matter? Ask yourself what the null hypothesis would and if it would matter.

8.8.6 Experiment: What happens to the `stayLength` slope?

Let's look at the coefficients for the three models involving `stayLength` as a predictor.

By itself.

term	estimate	std.error	statistic	p.value
(Intercept)	0.7443037	0.5538573	1.343855	0.1817359
stayLength	0.3742169	0.0563195	6.644530	0.0000000

With `cultureRatio`.

```
coeff.summary(riskLm)
```

term	estimate	std.error	statistic	p.value
(Intercept)	0.8054910	0.4877558	1.651423	0.1015044
stayLength	0.2754721	0.0524647	5.250615	0.0000007
cultureRatio	0.0564515	0.0097984	5.761279	0.0000001

And finally adding in `facilities`.

```
coeff.summary(riskierLm)
```

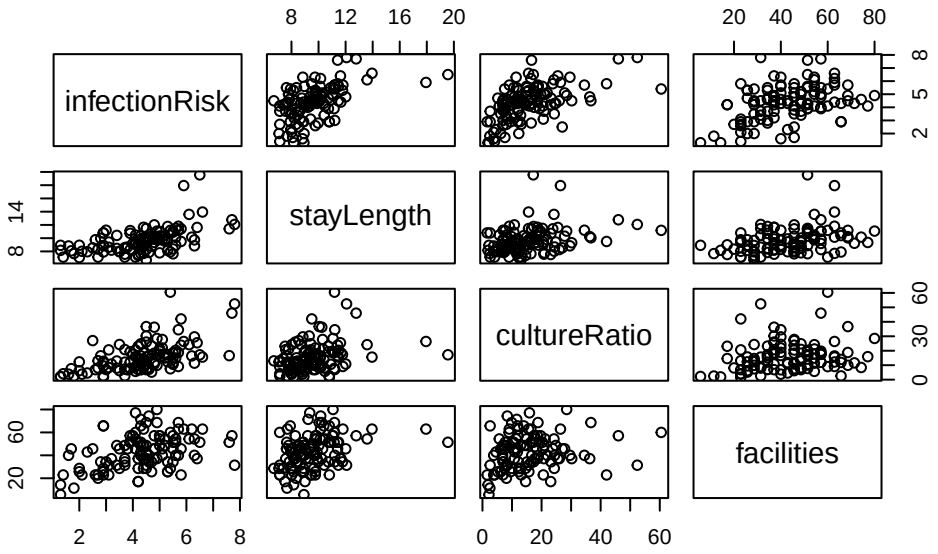
term	estimate	std.error	statistic	p.value
(Intercept)	0.4913323	0.4816361	1.020132	0.3099249
stayLength	0.2239075	0.0533656	4.195726	0.0000556
cultureRatio	0.0542000	0.0094793	5.717698	0.0000001
facilities	0.0196303	0.0064539	3.041603	0.0029482

8.9 Transformations

A simple way to check the relationship between the variables would be to use the `pairs()` function. The `pairs()` function produces scatterplots between *all* variables you specify.

The basic syntax is `pairs(formula, data)` where `formula` is the formula of the regression model you are considering `data` is the dataset you are using.

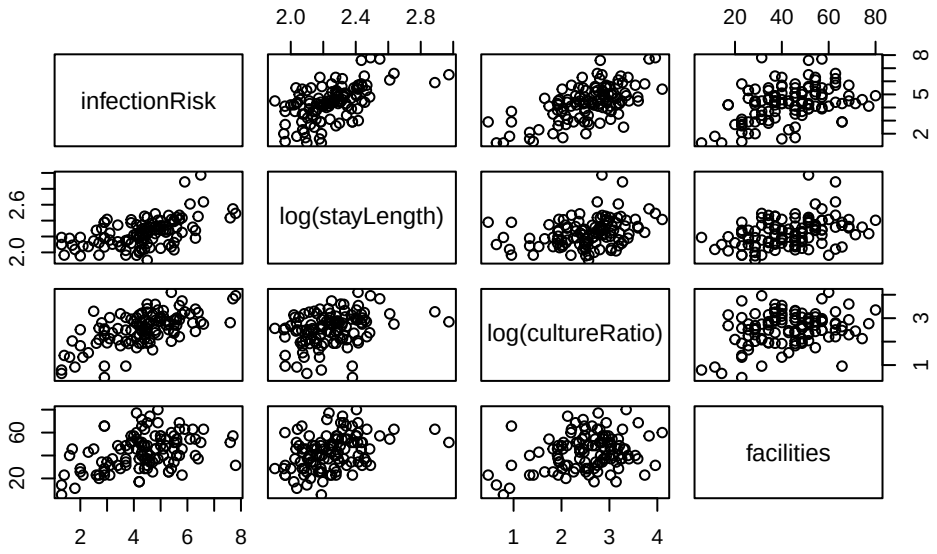
```
pairs(infectionRisk ~ stayLength + cultureRatio + facilities,
      senic)
```



This is a **scatterplot matrix**. Notice that the vertical axis determined by the variable named to the left or right of a plot, and horizontal axis is determined by the variable named above or below a plot.

8.9.1 Finding the right transformations

```
pairs(infectionRisk ~ log(stayLength) +
        log(cultureRatio) +
        facilities, senic)
```



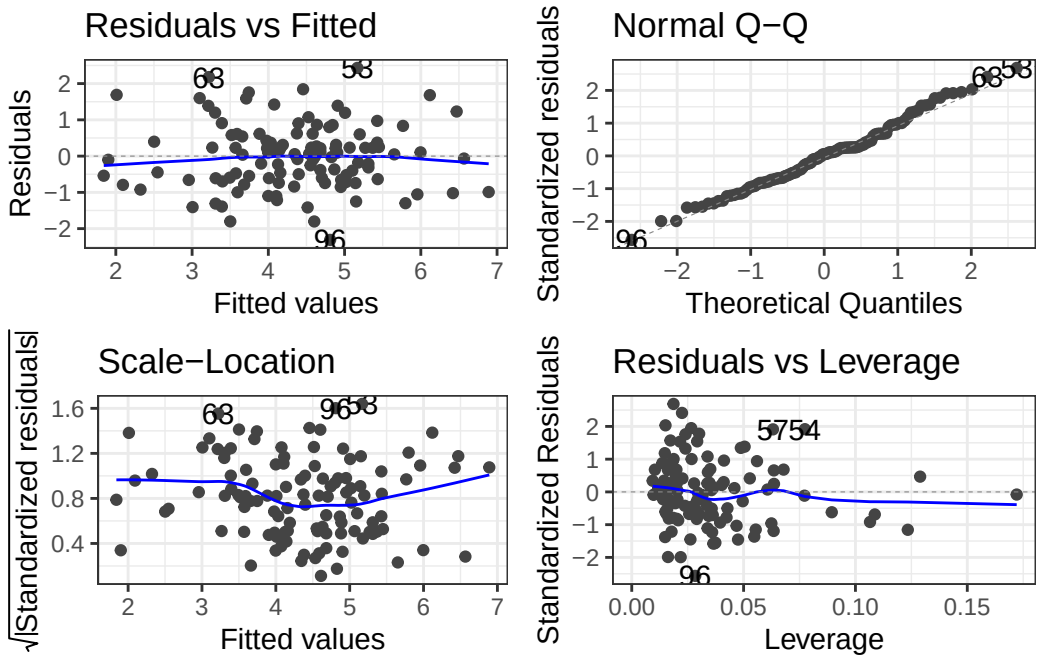
8.9.2 Incorporating them into the model

```
friskyLm <- lm(infectionRisk ~ log(stayLength) +
                 log(cultureRatio) +
                 facilities, senic)
summary(friskyLm)
```

```
Call:
lm(formula = infectionRisk ~ log(stayLength) + log(cultureRatio) +
    facilities, data = senic)

Residuals:
     Min       1Q   Median       3Q      Max
-2.31059 -0.63488  0.04808  0.54228  2.43225

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)       -4.255154   1.110040  -3.833 0.000212 ***
log(stayLength)    2.534644   0.540984   4.685 8.12e-06 ***
log(cultureRatio)  0.924657   0.134138   6.893 3.70e-10 ***
facilities         0.012736   0.006254   2.036 0.044147 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9138 on 109 degrees of freedom
Multiple R-squared:  0.548, Adjusted R-squared:  0.5355
F-statistic: 44.05 on 3 and 109 DF,  p-value: < 2.2e-16
```

8.9.3 Residuals!

```
autoplot(friskyLm)
```



Then you have to reinterpret everything, and so on and so on.

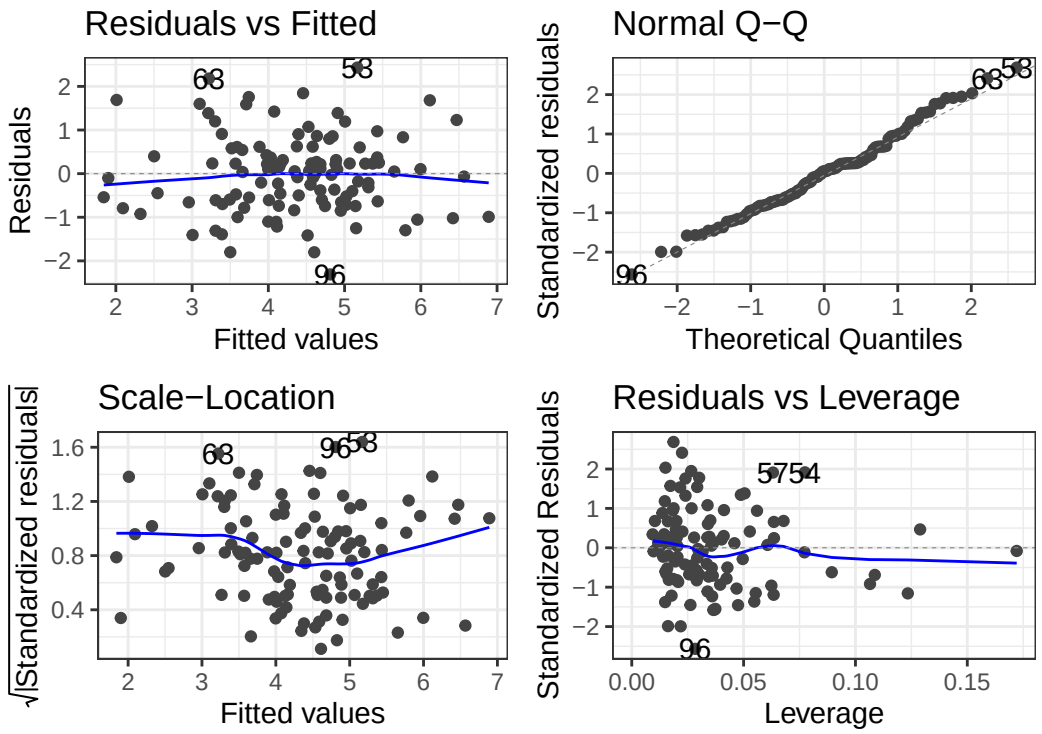
8.9.4 Which log?

```
friskyLm2 <- lm(infectionRisk ~ log2(stayLength) +
                  log2(cultureRatio) +
                  facilities, senic)
summary(friskyLm2)
```

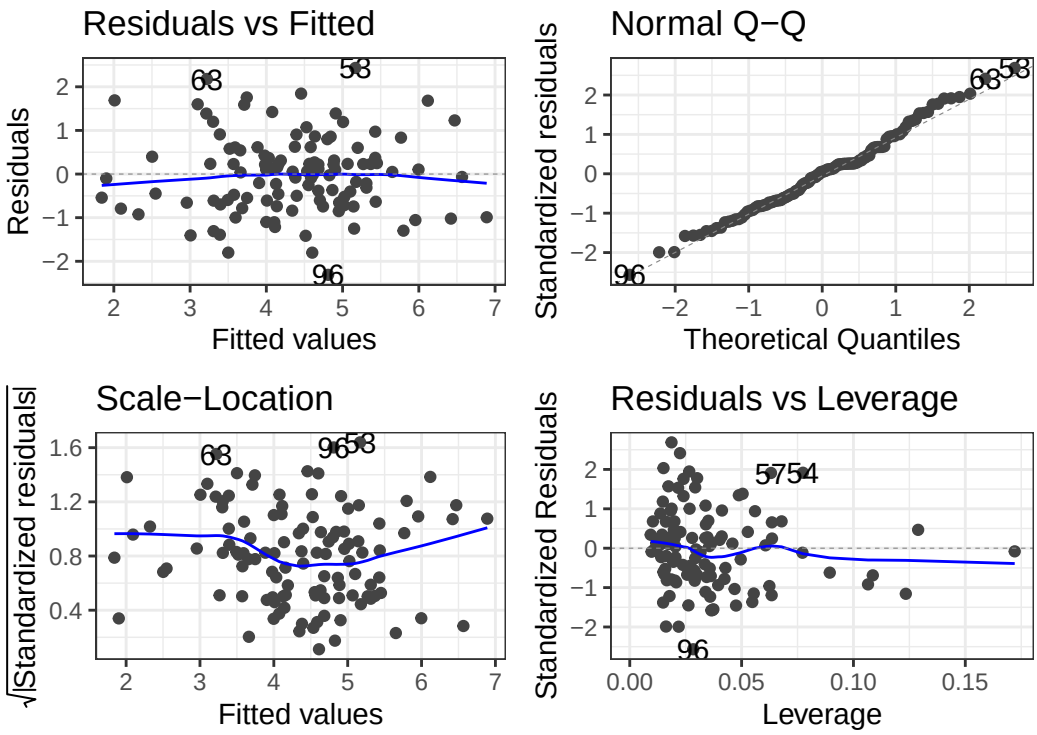
```
Call:
lm(formula = infectionRisk ~ log2(stayLength) + log2(cultureRatio) +
    facilities, data = senic)

Residuals:
     Min       1Q   Median       3Q      Max
-2.31059 -0.63488  0.04808  0.54228  2.43225

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)        -4.255154   1.110040  -3.833 0.000212 ***
log2(stayLength)    1.756881   0.374982   4.685 8.12e-06 ***
log2(cultureRatio)  0.640923   0.092977   6.893 3.70e-10 ***
facilities          0.012736   0.006254   2.036 0.044147 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9138 on 109 degrees of freedom
Multiple R-squared:  0.548, Adjusted R-squared:  0.5355
F-statistic: 44.05 on 3 and 109 DF,  p-value: < 2.2e-16
```

8.9.5 Can you spot the difference in residuals?

```
autoplot(friskyLm2)
```




```
autoplot(friskyLm)
```


Residuals vs Fitted
Residuals
Fitted values
68
58
96
Normal Q−Q
Standardized residuals
Theoretical Quantiles
68
58
96
Scale−Location
√|Standardized residuals|
Fitted values
68
96
58
Residuals vs Leverage
Standardized Residuals
Leverage
57
54
96

9 Variable Selection, Data Reduction, and Model Comparison

Here are some code chunks that setup this document.

```
# Here are the libraries I used
library(tidyverse) # standard
library(knitr) # need for a couple things to make knitted document to look nice
library(readr) # need to read in data
library(ggpubr) # allows for stat_cor in ggplots
library(ggfortify) # Needed for autoplot to work on lm()
library(gridExtra) # allows me to organize the graphs in a grid
library(car) # need for some regression stuff like vif
library(GGally)
library(Hmisc) # Needed for some visuals
library(rms) # needed for some data reduction tech.
library(pcaPP)
library(see)
library(performance)
```

```
# This changes the default theme of ggplot
old.theme <- theme_get()
theme_set(theme_bw())
```

9.1 Explainable statistical learning in public health for policy development: the case of real-world suicide data

We will work with data made available from this paper:

https://bmcmedresmethodol.biomedcentral.com/articles/10.1186/s12874-019-0796-7

If you want to go really in-depth of how you deal with data, this article goes into a lot of detail.

```
phe <- read_csv(here::here("datasets","phe.csv"))
```

9.1.1 Variables. A LOT!

Variables
2014 Suicide (age-standardised rate per 100,000 - outcome measure)
2013 Adult social care users who have as much social contact as they would like (% of adult social care users)
2013 Adults in treatment at specialist alcohol misuse services (rate per 1000 population)
2013 Adults in treatment at specialist drug misuse services (rate per 1000 population)
2013 Alcohol-related hospital admission (female) (directly standardised rate per 100,000 female population)
2013 Alcohol-related hospital admission (male) (directly standardised rate per 100,000 male population)
2013 Alcohol-related hospital admission (directly standardised rate per 100,000 population)
2013 Children in the youth justice system (rate per 1,000 aged 10–18)
2013 Children leaving care (rate per 10,000 < 18 population)
2013 Depression recorded prevalence (% of adults with a new diagnosis of depression who had a bio-psychosocial assessment)
2013 Domestic abuse incidents (rate per 1,000 population)
2013 Emergency hospital admissions for intentional self-harm (female) (directly age-standardised rate per 100,000 women)
2013 Emergency hospital admissions for intentional self-harm (male) (directly age-standardised rate per 100,000 men)
2013 Emergency hospital admissions for intentional self-harm (directly age-and-sex-standardised rate per 100,000)
2013 Looked after children (rate per 10,000 < 18 population)
2013 Self-reported well-being - high anxiety (% of people)
2013 Severe mental illness recorded prevalence (% of practice register [all ages])
2013 Social care mental health clients receiving services (rate per 100,000 population)
2013 Statutory homelessness (rate per 1000 households)

Variables
2013 Successful completion of alcohol treatment (% who do not represent within 6 months)
2013 Successful completion of drug treatment - non-opiate users (% who do not represent within 6 months)
2013 Successful completion of drug treatment - opiate users (% who do not represent within 6 months)
2013 Unemployment (% of working-age population)
2012 Adult carers who have as much social contact as they would like (18+ yrs) (% of 18+ carers)
2012 Adult carers who have as much social contact as they would like (all ages) (% of adult carers)
2011 Estimated prevalence of opiates and/or crack cocaine use (rate per 1,000 aged 15–64)
2011 Long-term health problems or disability (% of people whose day-to-day activities are limited by their health or disability)
2011 Marital breakup (% of adults whose current marital status is separated or divorced)
2011 Older people living alone (% of households occupied by a single person aged 65 or over)
2011 People living alone (% of all households occupied by a single person)
Mental Health Service users with crisis plans: % of people in contact with services with a crisis plan in place (end of quarter snapshot)
Older people
2011 Self-reported well-being - low happiness (% of people with a low happiness score)

9.2 How do we choose variables?

Objective: Find a way to predict suicide rates.

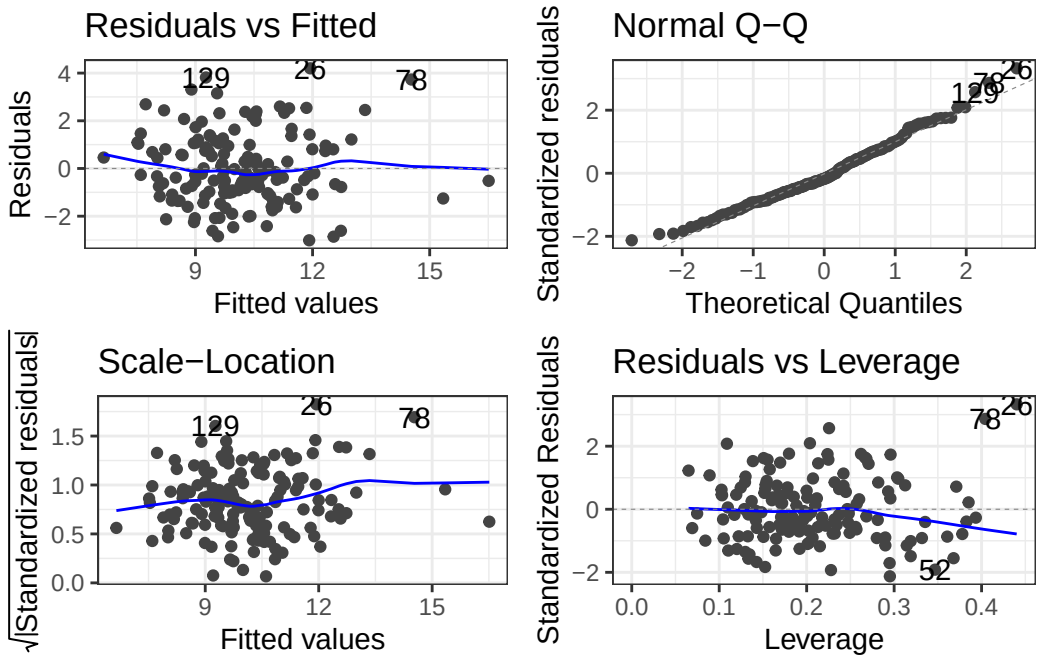
We could just use all of the predictors in a linear model.

```
fullLm <- lm(suicide_rate ~ ., phe)

summary(fullLm)
```

```

Call:
lm(formula = suicide_rate ~ ., data = phe)

Residuals:
    Min      1Q  Median      3Q     Max
-3.0118 -1.0447 -0.2702  0.9903  4.1993
```

```
Coefficients:
                               Estimate Std. Error t value Pr(>|t|)
(Intercept)                   2.0155877  3.5059381   0.575    0.566
children_youth_justice       -0.0235629  0.0247292  -0.953    0.343
adult_carers_isolated_18      0.0336685  0.0262501   1.283    0.202
adult_carers_isolated_all_ages -0.0226404  0.0424220  -0.534    0.595
adult_carers_not_isolated     0.3786174  0.2379446   1.591    0.114
alcohol_rx_18                -0.3001189  0.2515821  -1.193    0.235
alcohol_rx_all_ages          -0.0130574  0.0148093  -0.882    0.380
alcohol_admissions_f         -0.0060296  0.0127911  -0.471    0.638
alchol_admissions_m           0.0167345  0.0268020   0.624    0.534
alcohol_admissions_p         -0.0600782  0.0895445  -0.671    0.504
children_leaving_care         0.0216349  0.0288644   0.750    0.455
depression                   -0.1169805  0.1328876  -0.880    0.380
domestic_abuse                0.0269278  0.0419236   0.642    0.522
self_harm_female              0.0369915  0.0946913   0.391    0.697
self_harm_male                0.0318458  0.0944774   0.337    0.737
self_harm_persons            -0.0658812  0.1902674  -0.346    0.730
opiates                       0.2041520  0.1375203   1.485    0.140
lt_health_problems            0.0318830  0.1456081   0.219    0.827
lt_unepmloyment              -0.6342130  1.2588572  -0.504    0.615
looked_after_children         0.0174365  0.0146237   1.192    0.236
marital_breakup               0.1606036  0.1723378   0.932    0.353
old_pople_alone               0.3779967  0.4371635   0.865    0.389
alone                        -0.1004305  0.1386281  -0.724    0.470
self_reported_well_being      0.0605038  0.0681410   0.888    0.376
smi                           2.1708482  1.3401643   1.620    0.108
social_care_mh                0.0006815  0.0005270   1.293    0.199
homeless                     -0.1550536  0.1051424  -1.475    0.143
alcohol_rx                    0.0170587  0.0282631   0.604    0.547
drug_rx_non_opiate           -0.0408008  0.0271119  -1.505    0.135
drug_rx_opiate                0.1068925  0.0848400   1.260    0.210
unemployment                  0.0690206  0.2925910   0.236    0.814

Residual standard error: 1.687 on 118 degrees of freedom
Multiple R-squared:  0.5031,    Adjusted R-squared:  0.3767
F-statistic: 3.982 on 30 and 118 DF,  p-value: 3.823e-08
```

```
autoplot(fullLm)
```



If we are going by statistical significance, each predictor variable fails but the global F-test says that a linear model is working (sort of).

And that R^2 is pretty good for "prediciting" something related to human behavior.

Residuals look good.

Overall, this is bad!

You do not just use all the variables you have

Though there are exceptions: read Harrell's Regression Model Strategies (Chapter 4, Section 12, 4.12.1 - 4.12.3)

9.2.1 The scope of the problem

How many variables are there to use…? How well do they work?

We could try `cor(phe)` and see which variables are most correlated with `suicide_rate`. (You'll get a lot of output… that's $31 \times 31 = 961$ correlations)

I'll save you the pain and just show the correlations with just `suicide_rate`.

```
                              suicide_rate
children_youth_justice          0.05847330
adult_carers_isolated_18        0.24939518
```

adult_carers_isolated_all_ages	0.20369843
adult_carers_not_isolated	0.45228878
alcohol_rx_18	0.38997284
alcohol_rx_all_ages	0.33117982
alcohol_admissions_f	0.29906808
alchol_admissions_m	0.31868168
alcohol_admissions_p	0.11433511
children_leaving_care	0.36700280
depression	0.32509700
domestic_abuse	0.14835220
self_harm_female	0.49025136
self_harm_male	0.52004798
self_harm_persons	0.51686092
opiates	0.41195709
lt_health_problems	0.47825399
lt_unepmloyment	0.19643927
looked_after_children	0.51243274
marital_breakup	0.39971208
old_pople_alone	0.39870200
alone	0.31037882
self_reported_well_being	0.12949288
smi	0.18390513
social_care_mh	0.20125278
homeless	-0.32105739
alcohol_rx	-0.07513134
drug_rx_non_opiate	-0.14207504
drug_rx_opiate	-0.14345307
unemployment	0.18827138

9.2.2 Just use the best correlations?

These are a few of the strongest correlations.

	suicide_rate
self_harm_female	0.49025136
self_harm_male	0.52004798
self_harm_persons	0.51686092
looked_after_children	0.51243274

Let's make a model.

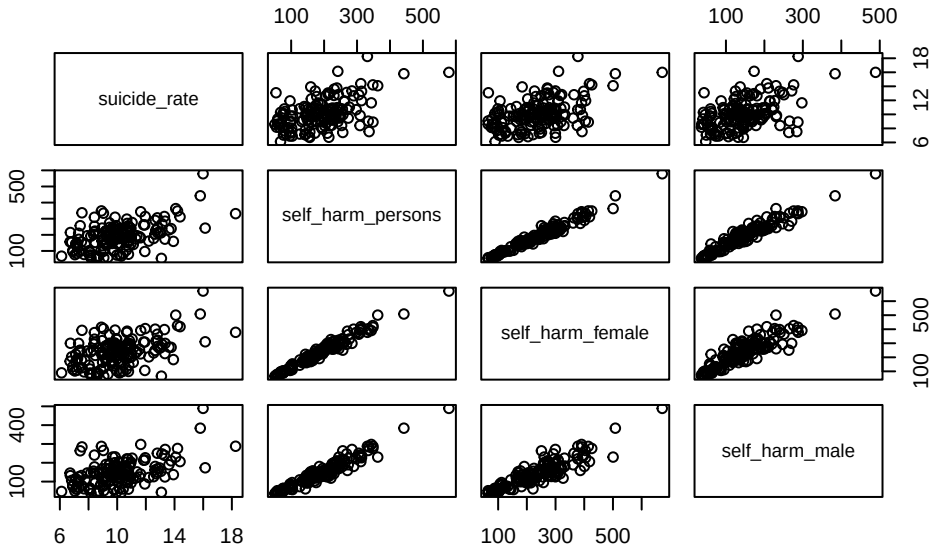
```r
badIdea <- lm(suicide_rate ~ self_harm_female + self_harm_male + self_harm_persons + looked_a

summary(badIdea)
```

```
Call:
lm(formula = suicide_rate ~ self_harm_female + self_harm_male +
    self_harm_persons + looked_after_children, data = phe)

Residuals:
    Min      1Q  Median      3Q     Max
-4.1974 -1.2196  0.0677  1.1319  6.4354

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)            6.521275   0.485989  13.419  < 2e-16 ***
self_harm_female       0.059297   0.087465   0.678 0.498889
self_harm_male         0.053601   0.087682   0.611 0.541954
self_harm_persons     -0.106747   0.175961  -0.607 0.545038
looked_after_children  0.031342   0.008442   3.713 0.000292 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.763 on 144 degrees of freedom
Multiple R-squared:  0.3379,    Adjusted R-squared:  0.3195
F-statistic: 18.37 on 4 and 144 DF,  p-value: 3.246e-12
```

What does the model have to say about how the self harm variables are related to suicide rate?

9.3 Multicollinearity

```r
pairs(suicide_rate ~ self_harm_persons + self_harm_female + self_harm_male, phe)
```




```
cor(phe$self_harm_female,phe$self_harm_male)
```

```
[1] 0.8865727
```

```
cor(phe$self_harm_persons,phe$self_harm_male)
```

```
[1] 0.9601094
```

```
cor(phe$self_harm_persons,phe$self_harm_female)
```

```
[1] 0.9804772
```

When we have predictor variables that are linearly related to each other, we have what is called **multi-collinearity**.

These variables essentially all contribute the same information over and over to the model. This makes a feedback loop that kicks everything around.

What about a model with just `self_harm_persons` (Why that one?) and `looked_after_children`?

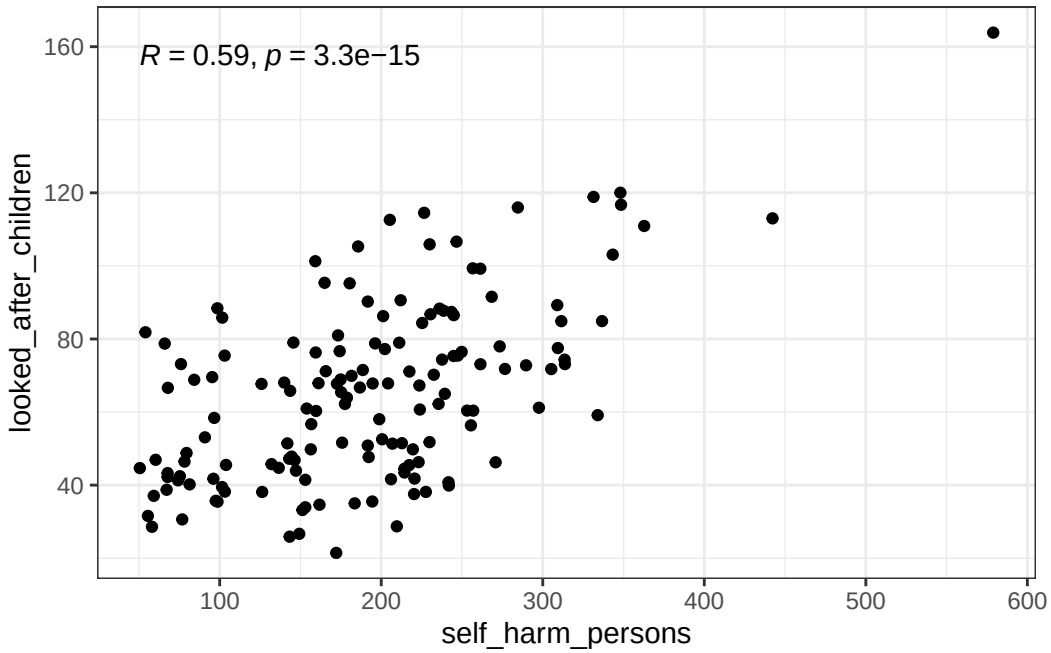
```r
notAsBadLm <- lm(suicide_rate ~ self_harm_persons + looked_after_children, phe)

summary(notAsBadLm)
```

```
Call:
lm(formula = suicide_rate ~ self_harm_persons + looked_after_children,
    data = phe)

Residuals:
    Min      1Q  Median      3Q     Max
-4.3450 -1.2323  0.0677  1.0792  6.2763

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)           6.704515   0.430054  15.590  < 2e-16 ***
self_harm_persons     0.008385   0.002124   3.947 0.000123 ***
looked_after_children 0.027803   0.007281   3.818 0.000198 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.756 on 146 degrees of freedom
Multiple R-squared:  0.3337,    Adjusted R-squared:  0.3246
F-statistic: 36.56 on 2 and 146 DF,  p-value: 1.344e-13
```

9.3.1 But what about self harm and looking after children?

Relations may not be as blaringly obvious as that. What about self harm and looking after children?

```r
ggplot(phe, aes(x = self_harm_persons, y = looked_after_children)) +
  geom_point() +
  stat_cor()
```



This isn't nearly as bad and wouldn't be considered a detrimental relation.

9.4 Measuring multi-collinearity

9.4.1 A linear model for the predictors

Linear relation between many predictor variables.

$$x_k = \beta_0^* + \beta_1^* x_1 + \beta_2^* x_2 + \cdots + \beta_p^* x_p + \epsilon^*$$

This is like any model and has it's own R^2 R_k^2.

If this R^2 is high, it means that a predictor variable is redundant.

There are a few ways to measure "high".

Some sources would say that $0.8 \leq R_k^2 \leq 0.9$ would be problematic, and you have extreme multicollinearity issues for anything higher.

9.4.2 Tolerance

Tolerance is $1 - R_k^2$.

That's it.

So 0.2 is the "problematic" cutoff and 0.9 is the "extreme" cutoff.

It may be referred to as TOL sometimes.

9.4.3 Variance Inflation Factors

The **Variance Inflation Factor** is:

$$VIF_k = \frac{1}{1 - R_k^2}$$

The VIF is considered "problematic" if greater than 5, and "extreme" if greater than 10.

9.4.4 Getting TOL or VIF: `performance` package

A nifty package that will help us deal with all the nasty bits of linear regression in `R` is the `performance` package. It is part of the `easystats` universe of packages.

To get tolerance and VIFs, create the model using `lm()`, and then plug it into the `check_collinearity` function.

- `check_collinearity(model)`

```
library(performance)
library(see)

# badidea was the model with all the self harm variables

check_collinearity(badIdea)
```

```
# Check for Multicollinearity

Low Correlation

                  Term  VIF          VIF 95% CI Increased SE Tolerance
 looked_after_children 2.04 [   1.66,     2.63]         1.43      0.49
 Tolerance 95% CI
```

```
     [0.38, 0.60]

High Correlation

              Term      VIF          VIF 95% CI Increased SE Tolerance
  self_harm_female  3811.50 [2809.27,  5171.42]        61.74  2.62e-04
    self_harm_male  1863.65 [1373.68,  2528.51]        43.17  5.37e-04
 self_harm_persons 10400.43 [7665.39, 14111.48]       101.98  9.61e-05
 Tolerance 95% CI
     [0.00, 0.00]
     [0.00, 0.00]
     [0.00, 0.00]
```

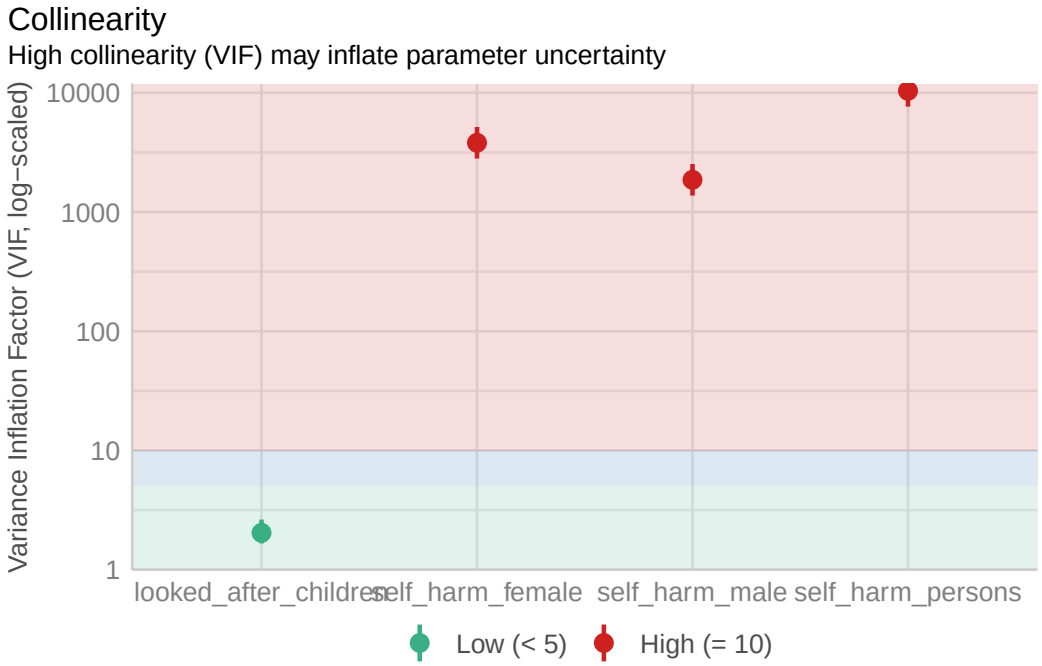
Versus

```
check_collinearity(notAsBadLm)
```

```
# Check for Multicollinearity

Low Correlation

                  Term  VIF   VIF 95% CI Increased SE Tolerance
     self_harm_persons 1.53 [1.29, 1.96]         1.24      0.65
 looked_after_children 1.53 [1.29, 1.96]         1.24      0.65
 Tolerance 95% CI
     [0.51, 0.78]
     [0.51, 0.78]
```

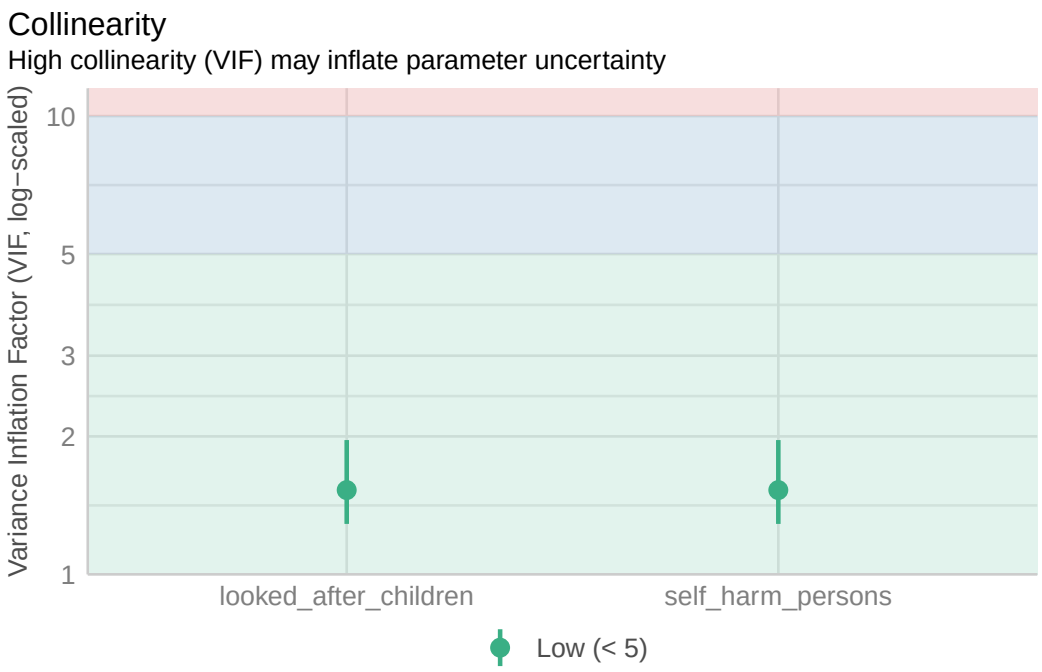
9.4.5 Plot `check_collinearity` checks

```
plot(check_collinearity(badIdea))
```



Versus

```
plot(check_collinearity(notAsBadLm))
```



9.4.6 VIF and Tolerance in the full model

- **You do not get rid of a variable because it has a small tolerance/large VIF!**
- You would investigate which variables are most correlated with eachother.
- You need to use your own reasoning to decide which variables to remove based on the correlations between predictors AND the response.

Tolerance and VIF only indicate that the overall model has a problem, not the individual variable.

```
check_collinearity(fullLm)
```

```
# Check for Multicollinearity

Low Correlation

                          Term  VIF           VIF 95% CI Increased SE
         children_youth_justice 1.48 [    1.29,     1.80]         1.22
       adult_carers_isolated_18 1.88 [    1.59,     2.30]         1.37
 adult_carers_isolated_all_ages 1.84 [    1.57,     2.26]         1.36
      adult_carers_not_isolated 3.11 [    2.55,     3.88]         1.76
           alcohol_admissions_p 2.05 [    1.73,     2.52]         1.43
          children_leaving_care 4.32 [    3.49,     5.42]         2.08
                     depression 2.73 [    2.25,     3.38]         1.65
                 domestic_abuse 1.90 [    1.61,     2.33]         1.38
                marital_breakup 2.36 [    1.97,     2.91]         1.54
                          alone 4.82 [    3.88,     6.06]         2.19
       self_reported_well_being 1.42 [    1.24,     1.72]         1.19
                            smi 3.62 [    2.94,     4.52]         1.90
                social_care_mh 1.29 [    1.15,     1.58]         1.14
                      homeless 3.11 [    2.55,     3.87]         1.76
                     alcohol_rx 3.27 [    2.68,     4.08]         1.81
             drug_rx_non_opiate 3.21 [    2.63,     4.00]         1.79
                 drug_rx_opiate 1.99 [    1.68,     2.44]         1.41
 Tolerance Tolerance 95% CI
      0.68     [0.56, 0.78]
      0.53     [0.43, 0.63]
      0.54     [0.44, 0.64]
      0.32     [0.26, 0.39]
      0.49     [0.40, 0.58]
      0.23     [0.18, 0.29]
```

0.37	[0.30, 0.44]
0.53	[0.43, 0.62]
0.42	[0.34, 0.51]
0.21	[0.16, 0.26]
0.71	[0.58, 0.81]
0.28	[0.22, 0.34]
0.77	[0.63, 0.87]
0.32	[0.26, 0.39]
0.31	[0.24, 0.37]
0.31	[0.25, 0.38]
0.50	[0.41, 0.60]

Moderate Correlation

Term	VIF	VIF 95% CI	Increased SE	Tolerance	Tolerance 95% CI
lt_unepmloyment	5.30	[4.26, 6.68]	2.30	0.19	[0.15, 0.23]
looked_after_children	6.68	[5.33, 8.45]	2.58	0.15	[0.12, 0.19]
unemployment	8.68	[6.89, 11.02]	2.95	0.12	[0.09, 0.15]

High Correlation

Term	VIF	VIF 95% CI	Increased SE	Tolerance	Tolerance 95% CI
alcohol_rx_18	18.48	[14.52, 23.59]	4.30	0.05	[0.04, 0.07]
alcohol_rx_all_ages	351.34	[273.84, 450.86]	18.74	2.85e-03	[0.00, 0.00]
alcohol_admissions_f	978.20	[762.20, 1255.50]	31.28	1.02e-03	[0.00, 0.00]
alchol_admissions_m	2345.59	[1827.47, 3010.68]	48.43	4.26e-04	[0.00, 0.00]
self_harm_female	4877.78	[3800.20, 6261.00]	69.84	2.05e-04	[0.00, 0.00]
self_harm_male	2362.51	[1840.66, 3032.40]	48.61	4.23e-04	[0.00, 0.00]
self_harm_persons	13277.66	[10344.20, 17043.10]	115.23	7.53e-05	[0.00, 0.00]
opiates	12.46	[9.83, 15.87]	3.53	0.08	
lt_health_problems	11.71	[9.25, 14.91]	3.42	0.09	
old_pople_alone	11.18	[8.84, 14.23]	3.34	0.09	

```
[0.00, 0.00]
[0.06, 0.10]
[0.07, 0.11]
[0.07, 0.11]
```

You have to jump through a lot of hoops and then make justifiable choices.

9.4.7 Detecting Multicollinearity

The following our indicators of multicollinearity:

1. "Significant" correlations between pairs of independent variables.
2. Non-significant t-tests for all or nearly all of the predictor variables, but a significant global F-test.
3. The coefficients have **opposite** signs than what would be expected (by logic or compared to the individual correlation with the response variable).
4. A single VIF of 10, more than one VIF of 5 or above. Several VIF 3 or over. Or corresponding tolerance values. Or corresponding R_k^2 values

The cutoffs for VIF, tolerance, or R_k^2 are **guidelines** not solid rules. Theory should always trump these more arbitrary statistical rules.

9.5 Variable screening the PHE data

The paper has various methods for variable selection to remove multicollinearity issues.

They don't really document it that well... Here's my best take.

```
# I created this dataset after going through things over and over
# I removed variables from the csv as I was working with it in R
# Lots of fun!

# also, I only use this dataset for demonstration so I don't have to type out
# all of the variables I left in the data into the formula.

pheRed <- read_csv(here::here("datasets",
                              'phe_reduced.csv'))

redModel <- lm(suicide_rate ~ ., pheRed)

check_collinearity(redModel)
```

```
# Check for Multicollinearity

Low Correlation

                          Term  VIF   VIF 95% CI Increased SE Tolerance
         children_youth_justice 1.13 [1.04, 1.51]         1.06      0.88
 adult_carers_isolated_all_ages 1.48 [1.27, 1.85]         1.22      0.68
           alcohol_admissions_p 1.78 [1.49, 2.23]         1.33      0.56
          children_leaving_care 2.58 [2.09, 3.30]         1.61      0.39
                     depression 2.21 [1.81, 2.80]         1.49      0.45
                 domestic_abuse 1.42 [1.23, 1.78]         1.19      0.70
              self_harm_persons 2.35 [1.92, 2.99]         1.53      0.43
                        opiates 2.14 [1.76, 2.70]         1.46      0.47
                marital_breakup 1.83 [1.53, 2.30]         1.35      0.55
                          alone 1.49 [1.28, 1.87]         1.22      0.67
       self_reported_well_being 1.24 [1.10, 1.57]         1.11      0.81
                  social_care_mh 1.17 [1.05, 1.51]         1.08      0.86
                       homeless 2.25 [1.84, 2.86]         1.50      0.44
                     alcohol_rx 1.43 [1.24, 1.80]         1.20      0.70
                  drug_rx_opiate 1.62 [1.37, 2.03]         1.27      0.62
 Tolerance 95% CI
     [0.66, 0.97]
     [0.54, 0.79]
     [0.45, 0.67]
     [0.30, 0.48]
     [0.36, 0.55]
     [0.56, 0.81]
     [0.33, 0.52]
     [0.37, 0.57]
     [0.43, 0.65]
     [0.54, 0.78]
     [0.64, 0.91]
     [0.66, 0.95]
     [0.35, 0.54]
     [0.56, 0.81]
     [0.49, 0.73]
```

- Some variables I removed because I could not find the correct definition.
- I am not an expert on this stuff so I had to do my best based on my mediocre knowledge of the subject matter, and trying to figure out what the authors of the paper were saying.
- When it is okay to stop removing variables is entirely dependent on the situation. I don't even have a guideline for it.

9.6 Next step: Choosing an actual model

Supposing the multicollinearity issue has been solved, how do we move on?

Do we use all the variables in the model?

```
summary(redModel)
```

```
Call:
lm(formula = suicide_rate ~ ., data = pheRed)

Residuals:
    Min      1Q  Median      3Q     Max
-3.8689 -1.1154 -0.0917  0.8901  4.2030

Coefficients:
                               Estimate Std. Error t value Pr(>|t|)
(Intercept)                   4.0832859  3.0285950   1.348   0.1799
children_youth_justice       -0.0136101  0.0220822  -0.616   0.5387
adult_carers_isolated_all_ages  0.0000509  0.0387482   0.001   0.9990
alcohol_admissions_p         -0.0623441  0.0849678  -0.734   0.4644
children_leaving_care         0.0422261  0.0227752   1.854   0.0659 .
depression                   -0.1622739  0.1220073  -1.330   0.1858
domestic_abuse                0.0247253  0.0369822   0.669   0.5049
self_harm_persons             0.0066734  0.0025813   2.585   0.0108 *
opiates                       0.1010386  0.0580642   1.740   0.0842 .
marital_breakup               0.2937180  0.1548091   1.897   0.0600 .
alone                         0.0336382  0.0786620   0.428   0.6696
self_reported_well_being      0.0345748  0.0650050   0.532   0.5957
social_care_mh                0.0007603  0.0005101   1.490   0.1385
homeless                     -0.2377671  0.0912485  -2.606   0.0102 *
alcohol_rx                   -0.0047326  0.0190770  -0.248   0.8045
drug_rx_opiate                0.0305590  0.0779842   0.392   0.6958
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.72 on 133 degrees of freedom
Multiple R-squared:  0.4176,    Adjusted R-squared:  0.3519
F-statistic: 6.358 on 15 and 133 DF,  p-value: 5.163e-10
```

- Well now at least some of the predictor's are significant.
- We should not just use all predictors since that ruins the generalizibility of the model.

9.6.0.1 The principle of parsimony: Models should have as few variables/parameters as possible while retaining adequacy.

9.7 Methods for model assessment

We have many variables to choose from, and it's really easy to make a model.

- Should we just pick the variables most highly correlated with suicide rate?
- What is our cutoff then?

	suicide_rate
children_youth_justice	0.05847330
adult_carers_isolated_all_ages	0.20369843
alcohol_admissions_p	0.11433511
children_leaving_care	0.36700280
depression	0.32509700
domestic_abuse	0.14835220
self_harm_persons	0.51686092
opiates	0.41195709
marital_breakup	0.39971208
alone	0.31037882
self_reported_well_being	0.12949288
social_care_mh	0.20125278
homeless	-0.32105739
alcohol_rx	-0.07513134
drug_rx_opiate	-0.14345307

```
cor1 <- lm(suicide_rate ~ self_harm_persons, phe)
cor2 <- lm(suicide_rate ~ self_harm_persons + opiates, phe)
cor3 <- lm(suicide_rate ~ self_harm_persons + opiates + marital_breakup, phe)
cor4 <- lm(suicide_rate ~ self_harm_persons + opiates + marital_breakup +
             children_leaving_care, phe)
cor5 <- lm(suicide_rate ~ self_harm_persons + opiates + marital_breakup +
             children_leaving_care + depression, phe)
cor6 <- lm(suicide_rate ~ self_harm_persons + opiates + marital_breakup +
             children_leaving_care + depression + homeless, phe)
```

- Is the last model the best?
- Should we add more variables?
- Different variables?

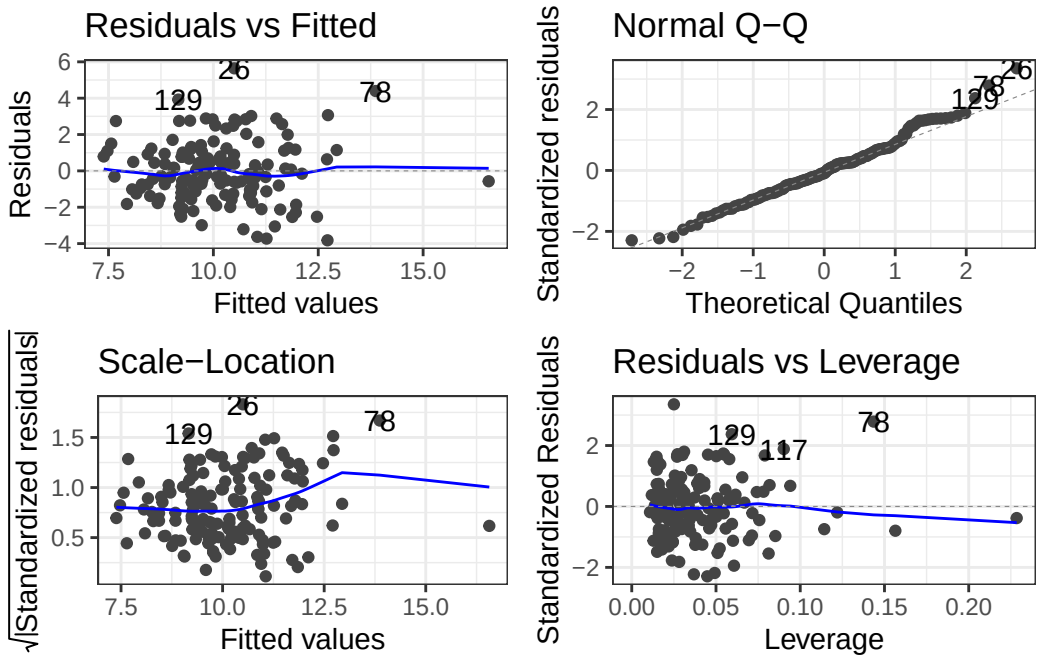
9.7.1 Just "significant" variables?

Lets use any of the variables that had a p-value below 0.1.

```
# Changing model name method since now we will be looking at several models

sig1 <- lm(suicide_rate ~ children_leaving_care + self_harm_persons + opiates + marital_breal

summary(sig1)
```

```
Call:
lm(formula = suicide_rate ~ children_leaving_care + self_harm_persons +
    opiates + marital_breakup + homeless, data = phe)

Residuals:
    Min      1Q  Median      3Q     Max
-3.8194 -1.1301 -0.1541  0.9383  5.6420

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)           4.855937   1.449776   3.349  0.00104 **
children_leaving_care 0.037193   0.019703   1.888  0.06109 .
self_harm_persons     0.005501   0.002307   2.385  0.01840 *
opiates               0.123432   0.050315   2.453  0.01536 *
marital_breakup       0.218171   0.134664   1.620  0.10741
homeless             -0.212926   0.075547  -2.818  0.00551 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.705 on 143 degrees of freedom
Multiple R-squared:  0.385, Adjusted R-squared:  0.3635
F-statistic: 17.91 on 5 and 143 DF,  p-value: 9.201e-14
```

```
autoplot(sig1)
```



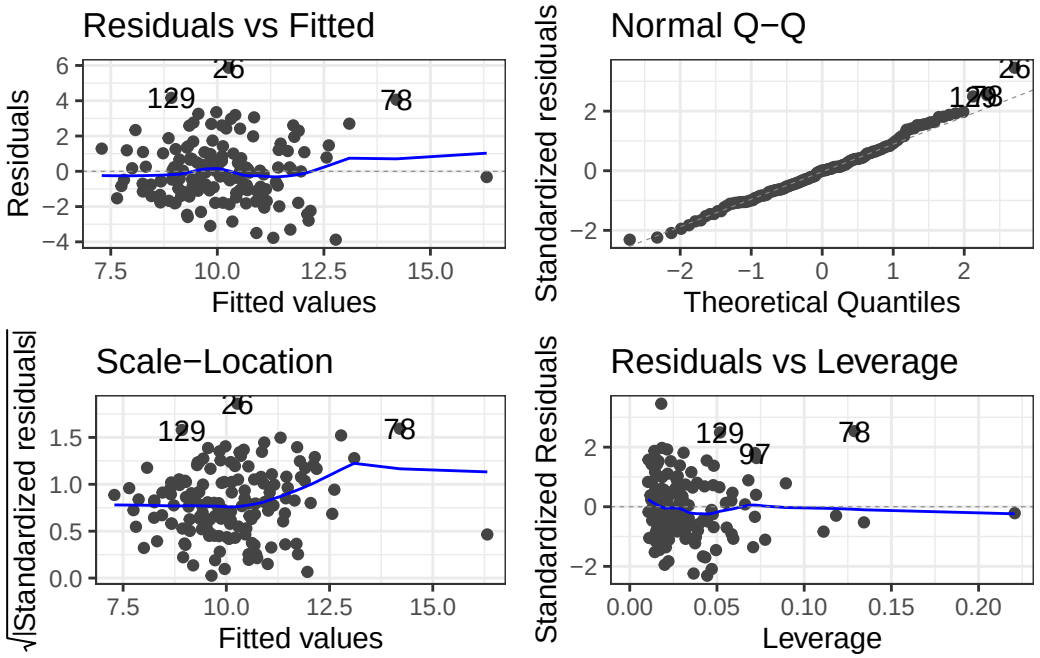
Wait... now marital breakup isn't significant?

```
# Changing model name method since now we will be looking at several models

sig2 <- lm(suicide_rate ~ children_leaving_care + self_harm_persons + opiates +
             homeless, phe)

summary(sig2)
```

```
Call:
lm(formula = suicide_rate ~ children_leaving_care + self_harm_persons +
    opiates + homeless, data = phe)

Residuals:
    Min      1Q  Median      3Q     Max
-3.8777 -1.1053  0.0011  1.0022  5.8723

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)            7.033367   0.546679  12.866  < 2e-16 ***
children_leaving_care  0.044142   0.019338   2.283  0.02392 *
self_harm_persons      0.006671   0.002203   3.028  0.00292 **
```

```
opiates                 0.121234   0.050580   2.397  0.01782 *
homeless               -0.227131   0.075459  -3.010  0.00309 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.714 on 144 degrees of freedom
Multiple R-squared:  0.3738,    Adjusted R-squared:  0.3564
F-statistic: 21.49 on 4 and 144 DF,  p-value: 6.473e-14
```

```
autoplot(sig2)
```



- Relying on "statistical" significance means you will end-up shooting yourself in the foot, metaphorically speaking, i.e., make a bad model.

9.7.2 R^2? (Don't use it to choose models).

Recall the definition of R^2: It is the proportion of variability in the response variable explained by your predictor variables in the linear model.

Higher R^2 is better, so maybe we should choose the model with the highest R^2?

- The full model has an R^2 of 0.5031.
- After whittling down some variables due to multicollinearity, `model1` has an R^2 of 0.385.

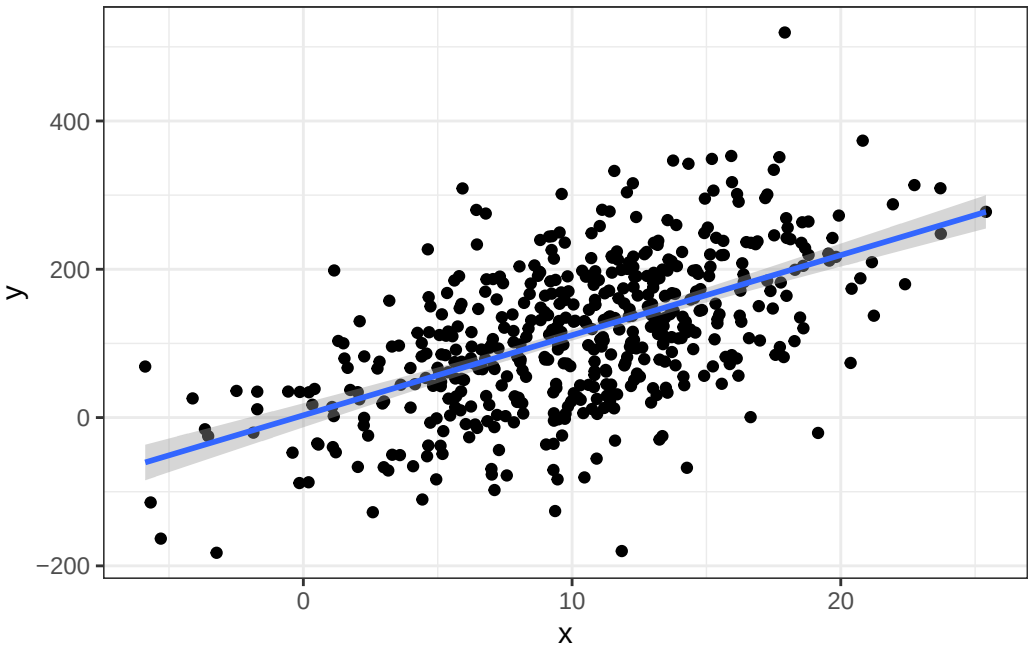
- `model2` has an R^2 of 0.3738.

So the full model is best! Right? All this stuff about variable elimination is pointless. I wasted hours of my life to get to this point…. No!

Never determine which is the best model via R^2.

- R^2 can be arbitrarily high when using many variables that may not be actually related to the response variable in any meaningful way.

```
n = 500
a = 10; b = 10

### MAKING DATA WITH A TRUE LINEAR MODEL
x <- rnorm(n, 10, 5)
y = a + b*x + rnorm(n,0 ,80)

data <- data.frame(y,x)

ggplot(data, aes(x,y)) +
  geom_point() +
  geom_smooth(method = 'lm')
```



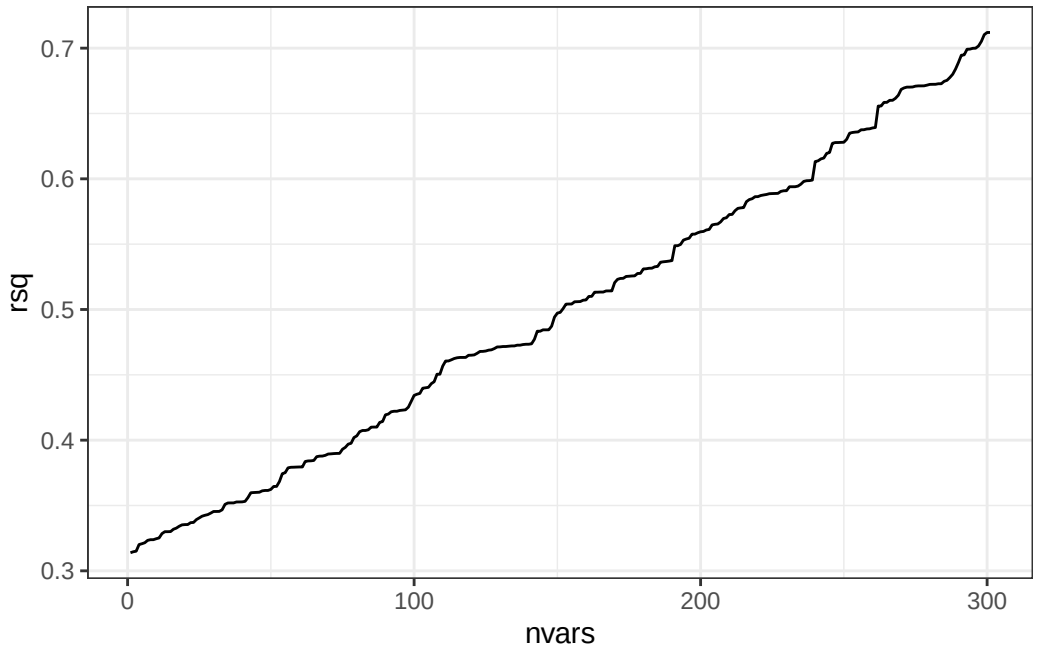
```
summary(lm(y ~ x, data))
```

```
Call:
lm(formula = y ~ x, data = data)

Residuals:
    Min      1Q  Median      3Q     Max
-311.10  -58.74    2.12   53.43  322.70

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   3.0884     8.2479   0.374    0.708
x            10.8010     0.7157  15.092   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 83.53 on 498 degrees of freedom
Multiple R-squared:  0.3138,    Adjusted R-squared:  0.3125
F-statistic: 227.8 on 1 and 498 DF,  p-value: < 2.2e-16
```

- The R-squared here *for the perfectly correct model* is 0.3138388
  - R-squared can be low even when you use the "correct" model.
- Now let's throw in some irrelevant data and graph what happens to R-squared.

```
# Extra worthless predictors

p = 300

extra.predictors <- data.frame(matrix(rnorm(n*p), nrow=n))

full.data <- data.frame(data, extra.predictors)# need code to combine true and extra data

rsq = rep(NA, p+1)

for(iter in 1:(p+1)){
form <-formula(paste('y ~ ', paste(names(full.data)[2:(iter+1)], collapse=' + ')))
fit.mdl <- lm(form, data=full.data)
rsq[iter] <- summary(fit.mdl)$r.squared # Get the rsq value for the model.
}
```

```
ggplot(data.frame(nvars=1:(p+1), rsq),aes(x=nvars,y=rsq)) +
geom_path()
```



- The beginning of the line on the left represents the R-squared value when using the one predictor variable that actually has any true relation with the response variable.
- When we add on additional predictors (useless ones), R-squared always increases.

9.7.3 Adjusted R^2 (More conservative)

$$R^2_{adj} = 1 - (1 - R^2)\frac{n-1}{n-p-1}$$

This is supposed to correct for the fact that R^2 always increases with a greater number of variables (penalizes for the complexity of the model).

Here's a table for most of the models we've discussed so far. It has both the adjusted and unadjusted form.

```
     model       Rsq    AdjRsq
1   fullLm 0.5030783 0.3767423
2 redModel 0.4176079 0.3519246
3     cor1 0.2671452 0.2621598
```

```
4     cor2 0.3275478 0.3183361
5     cor3 0.3476621 0.3341654
6     cor4 0.3508866 0.3328556
7     cor5 0.3514271 0.3287497
8     cor6 0.3943409 0.3687496
```

But… it doesn't necessarily. Don't trust it either.

9.7.4 Predicted R^2 (Even more conservative)

Predicted R-squared or as I will write R^2pred is another play on the same idea as R^2, except for it is based on taking out an observation, computing the regression model without it, then seeing how well the model predicts the left out value.

You can get it via the `olsrr` package using the `ols_pred_rsq()` function. The input is your model.

For instance, let us compare this to R^2 from the `performance` package.

```
r2(fullLm)
```

```
# R2 for Linear Regression
       R2: 0.503
  adj. R2: 0.377
```

```
library(olsrr)
ols_pred_rsq(fullLm)
```

```
[1] 0.1258262
```

That's a lot worse than the 0.503 for the unadjusted R-Squared

Here's a new table including predicted R-squared

```
     model       Rsq    AdjRsq   PredRsq
1   fullLm 0.5030783 0.3767423 0.1258262
2 redModel 0.4176079 0.3519246 0.2264297
3     cor1 0.2671452 0.2621598 0.2445699
4     cor2 0.3275478 0.3183361 0.2929179
5     cor3 0.3476621 0.3341654 0.3003604
6     cor4 0.3508866 0.3328556 0.2948558
7     cor5 0.3514271 0.3287497 0.2818233
8     cor6 0.3943409 0.3687496 0.3218613
```

- R^2 only tells how well the model works on the data you have (overfitting kills its utility).
- R^2_{adj} *tries* to make an attempt at saying how well the model would work on the wider population/future observations.
- R^2_{pred} is probably the best measure of how well your model will *actually* work with *new* data.
- **NONE** of these are perfect so use these tools with caution and care.

9.7.5 Akaike Information Criterion AIC

One last point of discussion will be the Akaike Information Criterion.

$$AIC = 2(p+1) + n\log(SSE/n) - C$$

There's a lot of theory behind this one and the Bayesian Information Criterion (BIC)

$$BIC = (p+1)\ln(n) + n\log(SSE/n) - C$$

This is where things get more confusing since AIC and BIC are numbers where SMALLER IS BETTER.

We will just be looking at AIC

```
     model       Rsq    AdjRsq   PredRsq      AIC
1   fullLm 0.5030783 0.3767423 0.1258262 183.0882
2 redModel 0.4176079 0.3519246 0.2264297 176.7362
3     cor1 0.2671452 0.2621598 0.2445699 182.9770
4     cor2 0.3275478 0.3183361 0.2929179 172.1605
5     cor3 0.3476621 0.3341654 0.3003604 169.6356
6     cor4 0.3508866 0.3328556 0.2948558 170.8973
7     cor5 0.3514271 0.3287497 0.2818233 172.7732
8     cor6 0.3943409 0.3687496 0.3218613 164.5730
```

Different measures will tell you different things.

Here R^2_{pred} and AIC agree that the `cor6` model is the best.

```
cor6 <- lm(suicide_rate ~ self_harm_persons + opiates + marital_breakup +
             children_leaving_care + depression + homeless, phe)
```

So is that the one we use?

It's not that simple.

9.8 Variable Selection Methods: Problems and Pitfalls

- Variable selection = methods for choosing which predictor variables to include in statistical models
- Focus on stepwise regression as a common but problematic approach
- Stepwise selection: automated process of adding/removing variables based on statistical significance (p-values and model fit)

9.8.1 Major Problems with Stepwise Selection

9.8.1.1 1. Biased R-squared Values

- R-squared values are inflated
- Makes model appear to explain more variation than it actually does
- Gives false confidence in model's predictive ability

9.8.1.2 2. Invalid Statistical Inference

- P-values are too small (not valid)
- Confidence intervals are too narrow
- Doesn't account for multiple testing problem
- Standard errors are biased low

9.8.1.3 3. Biased Regression Coefficients

- Coefficients are biased away from zero
- More likely to select variables with overestimated effects
- Selection process favors chance findings
- True effects may be much smaller than estimated

9.8.1.4 4. Model Instability

- Results highly dependent on sample
- Small changes in data can lead to different variables being selected
- Poor reproducibility
- Different samples likely to produce different models

9.8.2 Example: House Price Prediction

- Scenario: 20 potential predictors of house prices
- First sample might select:
  - Square footage
  - Number of bathrooms
  - Lot size
- Second sample might select completely different variables:
  - Age of house
  - Number of bedrooms
  - Distance to downtown
- Demonstrates instability of selection process

9.8.3 Better Alternatives

9.8.3.1 1. Theory-Driven Selection

- Use subject matter knowledge
- Select variables based on theoretical importance
- Include known important predictors regardless of significance

9.8.3.2 2. Include More Variables

- Keep theoretically important variables
- Don't eliminate based solely on statistical significance
- Better to include too many than too few important variables
  - Degrees of freedom permitting of course!

9.8.3.3 3. Alternative Dimension Reduction Methods

- Principal Components Analysis
- Regularization methods (LASSO, Ridge regression)
- Data reduction techniques that **don't** use outcome variable

9.8.4 Key Takeaways

- Avoid automated variable selection methods
- Don't let computational convenience override good statistical practice
- Complex but theoretically sound models preferred over overly simplified ones
- Statistical significance shouldn't be the sole criterion for variable inclusion

10 Prespecification of Predictor Complexity in Statistical Modeling

10.1 I. Introduction to Linear Relationships

- **Truly linear relationships are rare in real-world data**
- Notable exception: Same measurements at different timepoints
  - Example: Blood pressure before and after treatment
- Most relationships between predictors and outcomes are nonlinear
- Linear modeling often chosen due to data limitations, not reality

10.2 Problems with Post-Hoc Simplification

10.2.1 Common but Problematic Approaches:

1. Examining scatter plots
2. Checking descriptive statistics
3. Using informal assessments
4. Modifying model based on these observations

10.2.2 Key Issue:

- Creates "phantom degrees of freedom"
- Informal assessments use degrees of freedom not accounted for in:
  - Standard errors
  - P-values
  - R^2 values

10.3 The Prespecification Approach

10.3.1 Core Principles:

1. Decide on predictor complexity before examining relationships
2. Base decisions on:
    - Effective sample size
    - Prior knowledge
    - Expected predictor importance
3. Maintain decisions regardless of analysis results

10.3.2 Benefits:

- More reliable statistical inference
- Better representation of uncertainty
- Prevention of bias from data-driven simplification

10.4 Practical Implementation

10.4.1 Guidelines for Complexity:

- Allow more complex representations for:
    1. Stronger expected relationships
    2. Larger effective sample sizes
    3. More important predictors

10.4.2 Examples of Implementation:

- Using splines with more knots for important predictors
- Scaling complexity to sample size
- Matching complexity to theoretical importance

10.5 Validation and Testing

10.5.1 Allowed Practices:

- Graphing estimated relationships
- Performing nonlinearity tests
- Presenting results to readers

10.5.2 Important Rule:

- Maintain prespecified complexity even if simpler relationships appear adequate

10.6 The Directional Principle

10.6.1 Key Concepts:

1. Moving simple → complex
    - Degrees of freedom properly increase
    - Statistical tests maintain distribution
2. Moving complex → simple
    - Requires special adjustments
    - May compromise statistical validity

10.7 Importance and Impact

10.7.1 Benefits of Prespecification:

1. Prevents optimistic performance estimates
2. Maintains valid statistical inference
3. Provides reliable predictions
4. Avoids data-driven simplification bias

10.7.2 Trade-offs:

- May appear conservative
- Slight overfitting preferred to spurious precision

10.8 Summary

- Prespecify predictor complexity based on prior knowledge
- Avoid data-driven simplification
- Maintain statistical validity through consistent approach
- Better to slightly overfit than create spuriously precise estimates

10.9 Sample Size Requirements & Overfitting in Regression Models

10.9.1 Definition

- Overfitting occurs when model complexity exceeds data information content
- Results in:
  - Inflated measures of model performance (e.g., R^2)
  - Poor prediction on new data
  - Model fits noise rather than signal

10.9.1.1 Visual Example

```
Consider two models of the same data:
Simple model:    y ~ x
Overfit model:   y ~ x + x² + x³ + x  + ...
```

10.9.2 The m/15 Rule

10.9.2.1 Limiting Sample Size (m)

Type of Response	Limiting Sample Size (m)
Continuous	Total sample size (n)
Binary	min(n , n) *smaller group*
Ordinal (k categories)	n - $(1/n)\Sigma n^3$
Survival time	Number of events/failures

10.9.2.2 Basic Rule

- Number of parameters (p) should be $< m/15$
- Some situations require more conservative $p < m/20$
- Less conservative $p < m/10$ possible

10.9.3 Counting Parameters

10.9.3.1 Include ALL of these in your count:

1. Main predictor variables
2. Nonlinear terms
3. Interaction terms
4. Dummy variables for categorical predictors (k-1)

10.9.3.2 Example Parameter Count

```
Model components:
- Age (nonlinear, 3 knots) = 2 parameters
- Sex (binary) = 1 parameter
- Treatment (3 categories) = 2 parameters
- Age × Treatment interaction = 4 parameters
Total = 9 parameters
```

10.9.4 Special Considerations

10.9.4.1 Need More Conservative Ratios When:

- Predictors have narrow distributions
  - e.g., age range 30-45 years only
- Highly unbalanced categorical variables
  - e.g., 95% in one category
- Clustered measurements
- Small effect sizes expected

10.9.5 Practical Example

10.9.5.1 Binary Outcome Study

```
Study population:
- 1000 total patients
- 100 heart attacks
- 900 no heart attacks
```

```
Calculations:
m = min(100, 900) = 100
Maximum parameters = 100/15   6-7
```

10.9.6 Alternative Approaches

10.9.6.1 Other Methods to Assess/Prevent Overfitting:

1. Shrinkage estimates
2. Cross-validation
3. Bootstrap validation
4. Penalized regression methods

10.9.7 Sample Size for Variance Estimation

- Need ~70 observations for $\pm 20\%$ precision in σ estimate
- Affects all standard errors and p-values
- Critical for reliable inference

10.9.8 Key Takeaways

1. **Count Everything**: Include all terms in parameter count
2. **Be Conservative**: Use m/15 as starting point
3. **Consider Context**: Adjust for data peculiarities
4. **Validate**: Use shrinkage or cross-validation
5. **Simplify**: Prefer simpler models when in doubt

10.9.9 Practice Problems

1. Calculate maximum parameters for:
   - 500 patients, continuous outcome
   - 1000 patients, 150 events (survival)
   - 300 patients, 50/250 binary outcome
2. Count parameters in model:

   $y \sim age + age^2 + sex + treatment*race$
   where treatment has 3 levels and race has 4 levels

10.10 Shrinkage in Statistical Models: Understanding the Basics

10.10.1 Introduction

- Statistical models can suffer from **overfitting**
- Overfitting: model performs well on training data but poorly on new data
- Similar to memorizing test answers without understanding concepts
- Need methods to make models more reliable and generalizable

10.10.2 What is Shrinkage?

- Technique to prevent overfitting
- Makes model predictions more conservative and reliable
- Acts like a "leash" on model coefficients
- Helps handle regression to the mean

10.10.3 Example

- Study of 10 different medical treatments
- Some treatments appear very effective by chance
- When tested on new patients, effects usually less extreme
- Natural tendency for extreme results to move toward average
- This movement toward average is "shrinkage"

10.10.4 Key Shrinkage Methods

10.10.4.1 1. Ridge Regression

- Adds penalty to prevent **large** coefficients
- Characteristics:
  - Like rubber band pulling coefficients toward zero
  - Keeps all variables in model
  - Reduces size of effects
  - Ideal for correlated predictors

10.10.4.2 2. LASSO Regression

- Least Absolute Shrinkage and Selection Operator
- Characteristics:
  - Can force coefficients to exactly **zero**
  - Performs variable selection
  - Creates simpler models
  - Good for identifying important variables

10.10.5 Benefits of Shrinkage

1. More stable predictions
2. Better performance on new data
3. Protection against overfitting
4. More reliable assessment of variable importance

10.10.6 Key Takeaway

- Shrinkage represents statistical humility
- Prevents extreme predictions based on limited data
- Makes models more reliable and practical

10.11 Data Reduction Methods

10.11.1 Definition

- Process of reducing number of parameters in statistical models
- Focus on dimension reduction without using response variable Y
- "Unsupervised" approach to prevent overfitting

10.11.2 Purpose

- Improve model stability
- Reduce overfitting
- Maintain statistical inference validity
- Handle situations with many predictors relative to sample size

10.11.3 Redundancy Analysis

1. Core Concept
    - Identifies predictors well-predicted by other predictors
    - Removes redundant variables systematically
2. Implementation Process
    - Convert predictors to appropriate forms
        - Continuous → restricted cubic splines
        - Categorical → dummy variables
    - Use OLS for prediction
    - Remove highest R^2 predictors
    - Iterate until threshold reached

10.11.4 Variable Clustering

1. Purpose
    - Group related predictors
    - Identify independent dimensions
    - Simplify model structure
2. Methods
    - Statistical clustering with correlations
    - Principal component analysis (oblique rotation)
    - Hierarchical cluster analysis
3. Important Considerations
    - Use robust measures for skewed variables
    - Consider rank-based measures
    - Hoeffding's D for non-monotonic relationships

10.11.5 Variable Transformation and Scaling

1. Key Methods
    - Maximum Total Variance (MTV)
    - Maximum Generalized Variance (MGV)
2. Process Goals
    - Optimize variable transformations

- Maximize relationships between predictors
- Reduce complexity

3. Benefits
    - Fewer nonlinear terms needed
    - Better interpretability
    - More meaningful combinations

10.11.6 Simple Scoring of Variable Clusters

1. Approaches
    - First principal component
    - Weighted sums
    - Expert-assigned severity points
2. Common Applications
    - Binary predictor groups
    - Hierarchical scoring systems
    - Implementation-focused solutions

10.12 Implementation Guidelines

10.12.1 Best Practices

1. Prioritize subject matter knowledge
2. Validate without response variable
3. Use independent data for validation
4. Document reduction decisions
5. Balance simplicity vs. information

10.12.2 Recommended Workflow

1. Start with redundancy analysis
2. Apply variable clustering
3. Transform within clusters if needed
4. Create simple scores where appropriate
5. Validate each step

10.13 Key Considerations

10.13.1 Advantages

- Prevents overfitting
- Maintains statistical validity
- Improves model stability
- Enhances interpretability

10.13.2 Limitations

- Potential information loss
- Trade-off with complexity
- Need for validation

10.14 Discussion Points

10.14.1 Critical Questions

1. Choosing between methods
2. Determining optimal clusters
3. Balancing complexity and interpretability

10.14.2 Implementation Challenges

- Deciding on thresholds
- Handling mixed variable types
- Validating reduction decisions

10.14.3 Remarks

- Essential for stable modeling with many predictors
- Requires thoughtful method combination
- Focus on maintaining predictive power
- Ensure statistical validity

10.15 Data Reduction Techniques Examples

First, let's load the required packages and prepare our data:

```
library(Hmisc)
library(rms)
library(pcaPP)

# Read the data
df <- read.csv(here::here(
  "datasets",
  "phe.csv"))
```

- We have 149 rows of data
- We want to investigate the continuous outcome of `suicide_rate`
- m/15 rule: we can only have about 10 parameters in the model (we could be more liberal in loosen it to 15 variables)

10.15.1 1. Redundancy Analysis

Redundancy analysis helps identify predictors that can be well-predicted from other variables:

```
# Perform redundancy analysis
df2 = df %>% dplyr::select(-suicide_rate)
redun_result <- redun(~ .,
                      data=df, r2=0.6)

# Print results
redun_result
```

```
Redundancy Analysis

~.

n: 149  p: 31   nk: 3

Number of NAs:   0

Transformation of target variables forced to be linear
```

```
R-squared cutoff: 0.6   Type: ordinary

R^2 with which each variable can be predicted from all other variables:

        children_youth_justice       adult_carers_isolated_18
                         0.474                          0.634
adult_carers_isolated_all_ages      adult_carers_not_isolated
                         0.612                          0.769
                alcohol_rx_18           alcohol_rx_all_ages
                         0.965                          0.998
          alcohol_admissions_f           alchol_admissions_m
                         0.999                          1.000
          alcohol_admissions_p         children_leaving_care
                         0.686                          0.832
                    depression                domestic_abuse
                         0.738                          0.729
               self_harm_female                self_harm_male
                         1.000                          1.000
             self_harm_persons                        opiates
                         1.000                          0.952
           lt_health_problems              lt_unepmloyment
                         0.945                          0.876
         looked_after_children              marital_breakup
                         0.889                          0.706
               old_pople_alone                        alone
                         0.956                          0.906
      self_reported_well_being                          smi
                         0.520                          0.859
                social_care_mh                     homeless
                         0.513                          0.790
                    alcohol_rx             drug_rx_non_opiate
                         0.761                          0.787
                drug_rx_opiate                   suicide_rate
                         0.658                          0.639
                  unemployment
                         0.926

Rendundant variables:

self_harm_persons alchol_admissions_m alcohol_rx_all_ages alcohol_rx_18
old_pople_alone self_harm_male unemployment looked_after_children
lt_health_problems smi drug_rx_non_opiate opiates children_leaving_care
```

```
alcohol_admissions_f homeless self_harm_female


Predicted from variables:

children_youth_justice adult_carers_isolated_18
adult_carers_isolated_all_ages adult_carers_not_isolated
alcohol_admissions_p depression domestic_abuse lt_unepmloyment
marital_breakup alone self_reported_well_being social_care_mh
alcohol_rx drug_rx_opiate suicide_rate

        Variable Deleted   R^2
1      self_harm_persons 1.000
2    alchol_admissions_m 1.000
3    alcohol_rx_all_ages 0.978
4          alcohol_rx_18 0.960
5        old_pople_alone 0.950
6         self_harm_male 0.927
7           unemployment 0.896
8  looked_after_children 0.866
9     lt_health_problems 0.802
10                   smi 0.783
11    drug_rx_non_opiate 0.727
12               opiates 0.718
13 children_leaving_care 0.704
14  alcohol_admissions_f 0.684
15              homeless 0.663
16      self_harm_female 0.600
                                                                    R^2 after later deletions
1           1 1 1 1 0.987 0.987 0.986 0.986 0.986 0.985 0.985 0.984 0.981 0.981 0.625
2   0.997 0.997 0.997 0.996 0.996 0.996 0.996 0.996 0.996 0.996 0.996 0.7 0.696 0.665
3       0.978 0.978 0.977 0.977 0.976 0.974 0.973 0.972 0.972 0.972 0.717 0.713 0.677
4             0.959 0.955 0.952 0.951 0.951 0.947 0.947 0.792 0.787 0.755 0.751 0.742
5                   0.949 0.941 0.941 0.83 0.809 0.801 0.765 0.765 0.753 0.731 0.722
6                         0.925 0.923 0.921 0.92 0.917 0.915 0.912 0.895 0.891 0.636
7                               0.892 0.888 0.883 0.883 0.878 0.867 0.856 0.85 0.848
8                                     0.862 0.86 0.859 0.853 0.777 0.755 0.751 0.722
9                                           0.794 0.792 0.783 0.782 0.768 0.753 0.733
10                                                0.781 0.746 0.741 0.724 0.713 0.704
11                                                      0.724 0.718 0.717 0.716 0.702
12                                                            0.712 0.691 0.689 0.673
13                                                                  0.689 0.674 0.637
14                                                                        0.681 0.656
```

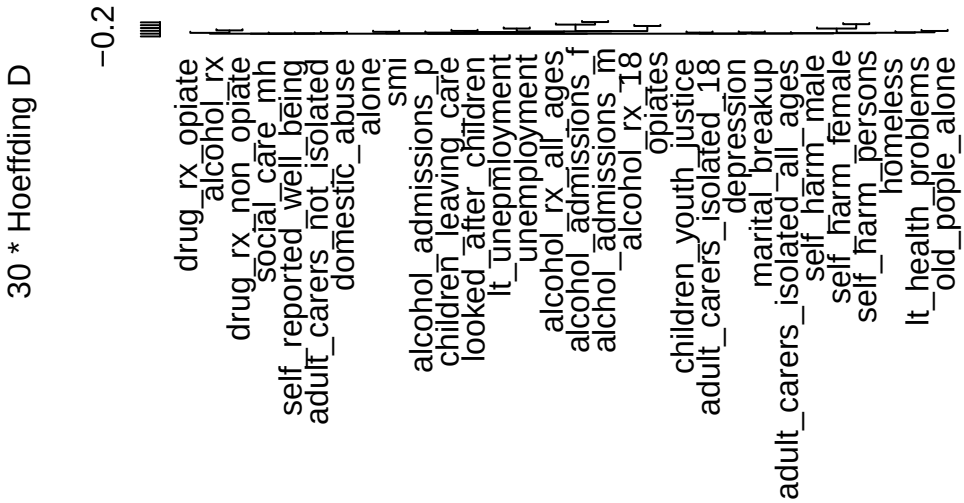
```
15                                                                    0.657
16
```

As with our section on multicollinarity, we can see that a number of variables are redundant. If we were to remove all these variables we would at least satisfy m/10 which is a bit more liberal than the m/15 guidance.

10.15.2 2. Variable Clustering

Let's perform hierarchical clustering on our variables to identify related groups:

```
# Perform variable clustering
vc <- varclus(~ .,
              data=df2, sim= 'hoeffding')

# Plot dendrogram
plot(vc)
```

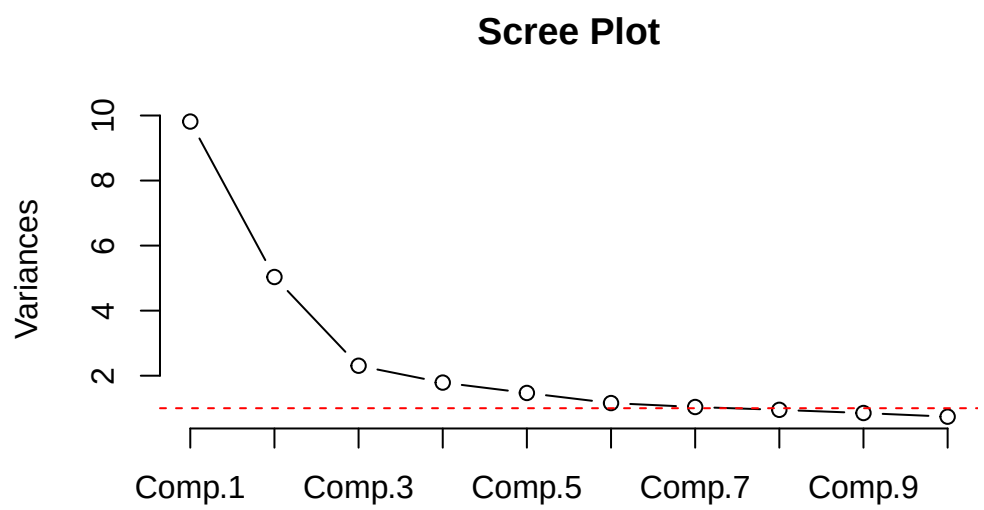


- We can see a lot of the variables clustering together

10.15.3 3. Principal Components Analysis

Let's examine the data structure using PCA:

```
# Perform PCA
pca_result <- princomp(df2, cor=TRUE)

# Scree plot
plot(pca_result, type="lines", main="Scree Plot")
abline(h=1, lty=2, col="red")  # Kaiser criterion line
```




```
# Print variance explained by first few components
summary(pca_result, loadings=TRUE, cutoff=0.3)
```

```
Importance of components:
                          Comp.1    Comp.2     Comp.3     Comp.4     Comp.5
Standard deviation     3.1327307 2.2436841 1.51857984 1.33713169 1.21224371
Proportion of Variance 0.3271334 0.1678039 0.07686949 0.05959737 0.04898449
Cumulative Proportion  0.3271334 0.4949373 0.57180682 0.63140419 0.68038869
                           Comp.6     Comp.7    Comp.8     Comp.9    Comp.10
Standard deviation     1.07625280 1.01836888 0.9747646 0.92320127 0.86023787
Proportion of Variance 0.03861067 0.03456917 0.0316722 0.02841002 0.02466697
```

```
Cumulative Proportion  0.71899936 0.75356853 0.7852407 0.81365075 0.83831772
                          Comp.11    Comp.12    Comp.13    Comp.14    Comp.15
Standard deviation     0.76164102 0.74326524 0.70575177 0.67895928 0.65072428
Proportion of Variance 0.01933657 0.01841477 0.01660285 0.01536619 0.01411474
Cumulative Proportion  0.85765429 0.87606907 0.89267192 0.90803811 0.92215284
                          Comp.16    Comp.17    Comp.18    Comp.19     Comp.20
Standard deviation     0.62502778 0.57314172 0.56636534 0.55357002 0.478567350
Proportion of Variance 0.01302199 0.01094971 0.01069232 0.01021466 0.007634224
Cumulative Proportion  0.93517484 0.94612455 0.95681687 0.96703153 0.974665755
                           Comp.21     Comp.22     Comp.23     Comp.24
Standard deviation     0.473155304 0.413788556 0.340046283 0.302671397
Proportion of Variance 0.007462531 0.005707366 0.003854382 0.003053666
Cumulative Proportion  0.982128287 0.987835652 0.991690035 0.994743701
                           Comp.25     Comp.26     Comp.27      Comp.28
Standard deviation     0.250796490 0.224198463 0.174227171 0.1176538993
Proportion of Variance 0.002096629 0.001675498 0.001011837 0.0004614147
Cumulative Proportion  0.996840330 0.998515828 0.999527665 0.9999890799
                            Comp.29      Comp.30
Standard deviation     1.669952e-02 6.980650e-03
Proportion of Variance 9.295803e-06 1.624316e-06
Cumulative Proportion  9.999984e-01 1.000000e+00

Loadings:
                             Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7
children_youth_justice                              0.378                0.580
adult_carers_isolated_18                            0.413                0.307
adult_carers_isolated_all_ages
adult_carers_not_isolated
alcohol_rx_18
alcohol_rx_all_ages
alcohol_admissions_f
alchol_admissions_m
alcohol_admissions_p
children_leaving_care
depression
domestic_abuse                                             0.342
self_harm_female
self_harm_male
self_harm_persons
opiates
lt_health_problems
lt_unepmloyment
looked_after_children
```

```
marital_breakup
old_pople_alone                    0.338
alone                                        -0.434
self_reported_well_being                                0.381
smi                                          -0.366
social_care_mh                                          0.488
homeless                           -0.325
alcohol_rx                                 -0.543
drug_rx_non_opiate                         -0.558
drug_rx_opiate                             -0.397
unemployment
                                 Comp.8 Comp.9 Comp.10 Comp.11 Comp.12 Comp.13
children_youth_justice
adult_carers_isolated_18
adult_carers_isolated_all_ages    0.351          -0.499  -0.301
adult_carers_not_isolated
alcohol_rx_18
alcohol_rx_all_ages
alcohol_admissions_f
alchol_admissions_m
alcohol_admissions_p              -0.378
children_leaving_care
depression                                                0.340   0.373
domestic_abuse                             -0.397         0.304
self_harm_female
self_harm_male
self_harm_persons
opiates                                                           0.335
lt_health_problems
lt_unepmloyment
looked_after_children
marital_breakup                   -0.316          -0.366  0.442
old_pople_alone                                                  -0.344
alone
self_reported_well_being           0.605 -0.374
smi
social_care_mh                    -0.618
homeless
alcohol_rx
drug_rx_non_opiate
drug_rx_opiate                                            0.312  -0.320
unemployment
                                 Comp.14 Comp.15 Comp.16 Comp.17 Comp.18 Comp.19
```

```
children_youth_justice                  0.376
adult_carers_isolated_18       -0.505  -0.378
adult_carers_isolated_all_ages                                  0.304
adult_carers_not_isolated      -0.390           0.510                  -0.459
alcohol_rx_18
alcohol_rx_all_ages
alcohol_admissions_f
alchol_admissions_m
alcohol_admissions_p                     0.376  -0.366
children_leaving_care                            0.393
depression                      -0.304           0.406          -0.387
domestic_abuse                                                   0.311
self_harm_female
self_harm_male
self_harm_persons
opiates                                         -0.305
lt_health_problems
lt_unepmloyment                         -0.448                  -0.334
looked_after_children
marital_breakup
old_pople_alone
alone
self_reported_well_being
smi
social_care_mh
homeless                                                        0.483
alcohol_rx
drug_rx_non_opiate                                              0.338
drug_rx_opiate                                                 -0.382
unemployment
                               Comp.20 Comp.21 Comp.22 Comp.23 Comp.24 Comp.25
children_youth_justice
adult_carers_isolated_18
adult_carers_isolated_all_ages
adult_carers_not_isolated
alcohol_rx_18
alcohol_rx_all_ages
alcohol_admissions_f
alchol_admissions_m
alcohol_admissions_p
children_leaving_care                                   -0.469
depression
domestic_abuse
```

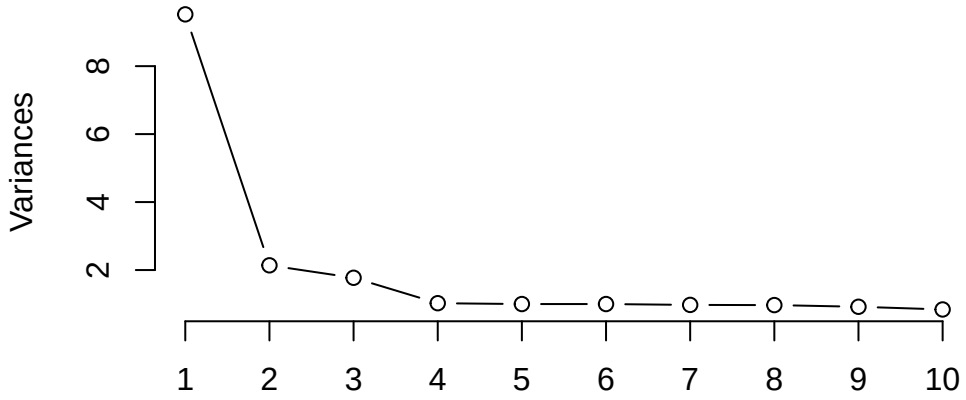
```
self_harm_female                                                  0.495
self_harm_male                                                   -0.668
self_harm_persons
opiates
lt_health_problems              0.423
lt_unepmloyment                                          0.422
looked_after_children                             0.634  0.429
marital_breakup                -0.370
old_pople_alone                 0.307
alone                                  -0.441
self_reported_well_being
smi                                     0.561  -0.314
social_care_mh
homeless                        0.349
alcohol_rx                                      0.525
drug_rx_non_opiate                     -0.316  -0.454
drug_rx_opiate
unemployment                                            -0.659
                               Comp.26 Comp.27 Comp.28 Comp.29 Comp.30
children_youth_justice
adult_carers_isolated_18
adult_carers_isolated_all_ages
adult_carers_not_isolated
alcohol_rx_18                   0.342  -0.625
alcohol_rx_all_ages                            -0.760
alcohol_admissions_f                            0.590   0.513
alchol_admissions_m                                    -0.806
alcohol_admissions_p
children_leaving_care
depression
domestic_abuse
self_harm_female                                         0.487
self_harm_male                         -0.353            0.338
self_harm_persons                                       -0.804
opiates                        -0.379   0.435
lt_health_problems              0.479
lt_unepmloyment
looked_after_children
marital_breakup
old_pople_alone                -0.488
alone
self_reported_well_being
smi
```

```
social_care_mh
homeless
alcohol_rx
drug_rx_non_opiate
drug_rx_opiate
unemployment                    -0.312
```

10.15.4 4. Sparse Principal Components Analysis

Using pcaPP for robust sparse PCA.

```
# Perform sparse PCA
sparse_pca <- sPCAgrid(df2, k=10, method = 'sd' ,
center =mean , scale =sd , scores =TRUE ,
maxiter =10)

# Plot variance explained
plot(sparse_pca,type = 'lines' , main= ' ')
```




```
# Print loadings
print(sparse_pca$loadings)
```

```
Loadings:
                            Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7
children_youth_justice                                              1.000
adult_carers_isolated_18
adult_carers_isolated_all_ages       0.464
adult_carers_not_isolated    0.252
alcohol_rx_18                0.295
alcohol_rx_all_ages          0.305
alcohol_admissions_f         0.295
alchol_admissions_m          0.303
alcohol_admissions_p         0.161
children_leaving_care        0.248
depression                   0.111
domestic_abuse               0.106                                  0.791
self_harm_female             0.131         -0.264
self_harm_male               0.222
self_harm_persons            0.176
opiates                      0.278
lt_health_problems           0.210
lt_unepmloyment              0.226
looked_after_children        0.300
marital_breakup              0.113
old_pople_alone                     0.636
alone                        0.122                                 -0.607
self_reported_well_being                  0.965
smi                          0.122
social_care_mh                                      1.000
homeless                           -0.613
alcohol_rx                                0.707
drug_rx_non_opiate                        0.707
drug_rx_opiate
unemployment                 0.234
                            Comp.8 Comp.9 Comp.10
children_youth_justice
adult_carers_isolated_18
adult_carers_isolated_all_ages
adult_carers_not_isolated
alcohol_rx_18
alcohol_rx_all_ages
alcohol_admissions_f
alchol_admissions_m
alcohol_admissions_p                      -0.574
```

```
children_leaving_care
depression                       0.903
domestic_abuse
self_harm_female
self_harm_male
self_harm_persons                       0.444
opiates
lt_health_problems
lt_unepmloyment
looked_after_children
marital_breakup                                  0.819
old_pople_alone
alone
self_reported_well_being
smi
social_care_mh
homeless
alcohol_rx
drug_rx_non_opiate
drug_rx_opiate                          0.896
unemployment                    -0.426

               Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8 Comp.9
SS loadings     1.000  1.000  1.000  1.000  1.000  1.000  1.000  1.000  1.000
Proportion Var  0.033  0.033  0.033  0.033  0.033  0.033  0.033  0.033  0.033
Cumulative Var  0.033  0.067  0.100  0.133  0.167  0.200  0.233  0.267  0.300
               Comp.10
SS loadings      1.000
Proportion Var   0.033
Cumulative Var   0.333
```

Like are variable clutstering, we can see that *lots* of variables load onto the first principal component. We could add component 1 to our regression model and keep the rest of the variables (not those loaded on comp. 1).

However, let's instead use stepwise regression *only on the preditors* in combination with sparse PCA to select variables for 10 components in our model.

- Stepwise here tells us *which* predictors that load onto each PC are needed to predict that pc

```r
spca1 <- princmp(df2, sw = TRUE ,
                 k = 10,
                 kapprox = 10,
                 method = "sparse",
                 cor = TRUE,
                 nvmax = 30)

print(spca1)
```

```
Sparse Principal Components Analysis

Stepwise Approximations to PCs With Cumulative R^2

PC 1
alchol_admissions_m (0.975) + self_harm_persons (0.999) +
looked_after_children (1)

PC 2
adult_carers_isolated_all_ages (0.759) + homeless (0.978) +
old_pople_alone (1)

PC 3
drug_rx_non_opiate (0.89) + alcohol_rx (1)

PC 4
self_harm_female (1)

PC 5
adult_carers_isolated_18 (0.861) + self_harm_female (1)

PC 6
self_harm_female (0.993) + self_reported_well_being (1)

PC 7
social_care_mh (1)

PC 8
children_youth_justice (1)

PC 9
domestic_abuse (0.872) + alone (1)
```

```
PC 10
depression (0.875) + unemployment (0.999) + old_pople_alone (1)
```

We get 10 components, some with many predictors, some with only one.

10.15.5 Put it all together

Using theory, we could decide to create cluster scores for some of these components, while others with only 2 variables we might want to just eliminate one of the variables.

For those that we want to have a composite score, we can use the first principal component of their scores. Or, if we want to use this scoring outside of this context we can get a regression model on the first PC.

Let me demonstrate with our first component from the sparse PCA.

```
# Get pc scores for the first sparse PC variables
pc1_scores = princmp(df2[,c(names(spca1$sw[[1]]))],
                     method = "regular",
                     cor = TRUE)$scores[,1]
df2$pc1_scores = pc1_scores

mod1 = lm(pc1_scores~alchol_admissions_m + self_harm_persons , data = df2)

parameters::parameters(mod1) |> print_md()
```

Parameter	Coefficient	SE	95% CI	t(146)	p
(Intercept)	-6.53	0.18	(-6.88, -6.17)	-36.38	< .001
alchol admissions m	3.63e-03	1.47e-04	(3.34e-03, 3.92e-03)	24.70	< .001
self harm persons	9.09e-03	4.38e-04	(8.22e-03, 9.95e-03)	20.75	< .001

Now, we could report those regression coefficients to get a pretty reasonable estimate of the scoring for that component of the model.

Now, let's get the second and third scores as well.

```
# Get pc scores for the second sparse PC variables
pc2_scores = princmp(df2[,c(names(spca1$sw[[2]]))],
                     method = "regular",
                     cor = TRUE)$scores[,1]
```

```r
df2$pc2_scores = pc2_scores

pc3_scores = princmp(df2[,c(names(spca1$sw[[3]]))],
                     method = "regular",
                     cor = TRUE)$scores[,1]

df2$pc3_scores = pc3_scores
```

For the rest, we will simplify things a bit and just use the first variable of the remaining 10 sparse PCA components.

Let's now fit a model based using our 3 PCs and the 7 other variables we selected.

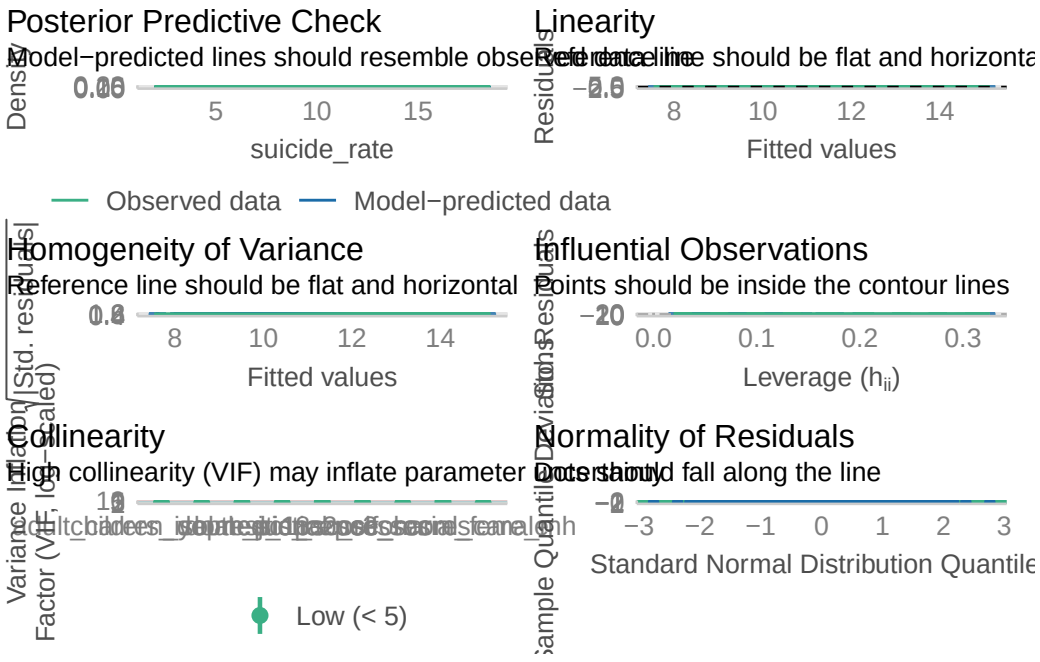
```r
df_all = df2 %>%
  select(pc1_scores, pc2_scores, pc3_scores,
         self_harm_female,
         adult_carers_isolated_18,
         self_harm_female,
         social_care_mh,
         children_youth_justice,
         domestic_abuse,
         depression)

df_all$suicide_rate = df$suicide_rate

# 10 predictors, 1 outcome, 149 observations
model_all = lm(
  data = df_all,
  suicide_rate ~ .
)

summary(model_all)
```

```
Call:
lm(formula = suicide_rate ~ ., data = df_all)

Residuals:
    Min      1Q  Median      3Q     Max
-4.2067 -1.1131 -0.1051  0.9371  4.1430

Coefficients:
```

```
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)               9.6296409  1.6223906   5.935 2.22e-08 ***
pc1_scores                0.5747571  0.1555263   3.696 0.000315 ***
pc2_scores                0.3436984  0.1388871   2.475 0.014539 *
pc3_scores               -0.1009508  0.1084578  -0.931 0.353579
self_harm_female          0.0025597  0.0024544   1.043 0.298819
adult_carers_isolated_18  0.0239721  0.0243409   0.985 0.326408
social_care_mh            0.0010556  0.0004836   2.183 0.030729 *
children_youth_justice   -0.0310969  0.0241586  -1.287 0.200165
domestic_abuse            0.0266173  0.0359734   0.740 0.460599
depression               -0.1202582  0.1114831  -1.079 0.282584
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.729 on 139 degrees of freedom
Multiple R-squared:  0.3853,    Adjusted R-squared:  0.3455
F-statistic: 9.681 on 9 and 139 DF,  p-value: 2.122e-11
```

Now let's check our assumptions:

```
check_model(model_all)
```



10.15.6 Analysis Summary

1. From the redundancy analysis, we can identify variables that are highly predictable from others, helping reduce dimensionality while maintaining information.
2. The variable clustering dendrogram shows natural groupings in our data, which can guide feature selection or creation of composite scores.
3. The PCA results show how many components are needed to explain a certain percentage of variance in the data.
4. Sparse PCA provides a more interpretable solution by forcing some loadings to zero while maintaining most of the explained variance.

10.15.7 Recommendations for Data Reduction

Based on these analyses, we can recommend:

1. Consider combining highly correlated variables within the same cluster into composite scores
2. Use the first few principal components if dimension reduction is needed while maintaining maximum variance
3. For interpretability, consider using the sparse PCA solution which provides clearer variable groupings
4. Remove redundant variables identified in the redundancy analysis on an as needed basis. Justify your choices!

These techniques provide different perspectives on data reduction, and the choice depends on the specific needs of the analysis (interpretability, variance preservation, or prediction accuracy).

11 Outliers and Influential Observations

Here are some code chunks that setup this chapter.

```
# Here are the libraries I used
library(tidyverse) # standard
# need for a couple things to make knitted document to look nice
library(knitr)
# need to read in data
library(readr)
# allows for stat_cor in ggplots
library(ggpubr)
# Needed for autoplot to work on lm()
library(ggfortify)
# allows me to organize the graphs in a grid
library(gridExtra)
 # need for some regression stuff like vif
library(car)
library(GGally)
```

```
# This changes the default theme of ggplot
old.theme <- theme_get()
theme_set(theme_bw())
```

11.1 Explainable statistical learning in public health for policy development: the case of real-world suicide data

We will work with data made available from this paper:

https://bmcmedresmethodol.biomedcentral.com/articles/10.1186/s12874-019-0796-7

If you want to go really in-depth of how you deal with data, this article goes into a lot of detail.

```
pheRed <- read_csv(here::here("datasets",
                              "phe_reduced.csv"))
```

11.1.1 Variables. A LOT!

```
colnames(pheRed)
```

```
 [1] "children_youth_justice"         "adult_carers_isolated_all_ages"
 [3] "alcohol_admissions_p"           "children_leaving_care"
 [5] "depression"                     "domestic_abuse"
 [7] "self_harm_persons"              "opiates"
 [9] "marital_breakup"                "alone"
[11] "self_reported_well_being"       "social_care_mh"
[13] "homeless"                       "alcohol_rx"
[15] "drug_rx_opiate"                 "suicide_rate"
```

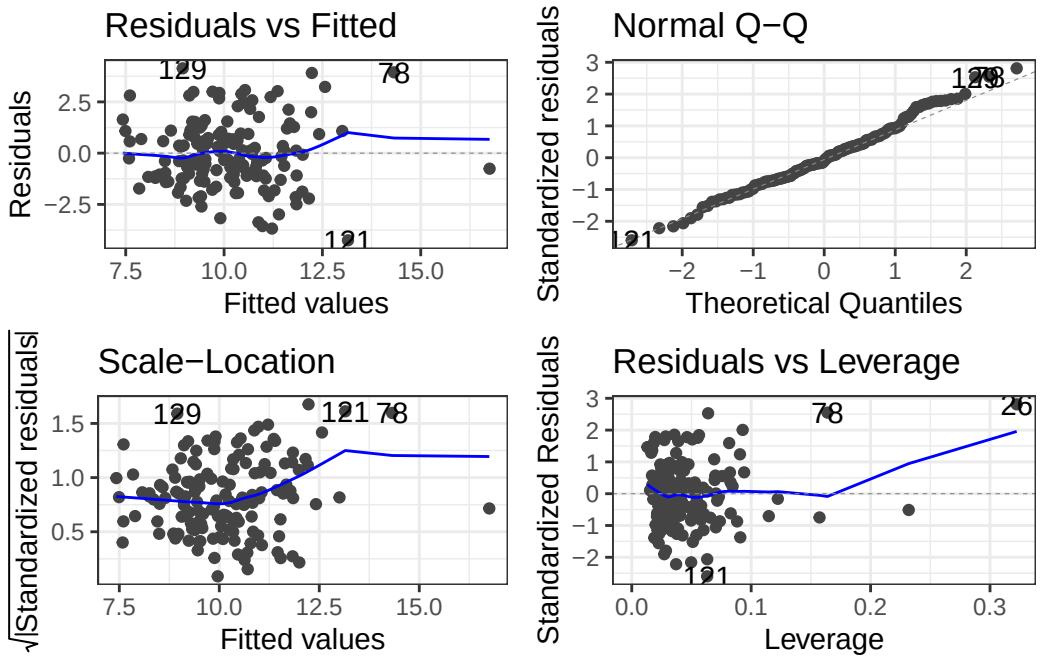
11.2 We have a model!

Using stepwise regression, we (supposedly) got a "good" model for "predicting" suicide rates:

```
fullModel <- lm(suicide_rate ~ ., pheRed)

model <- step(fullModel, trace = 0)

summary(model)
```

```
Call:
lm(formula = suicide_rate ~ children_leaving_care + self_harm_persons +
    opiates + marital_breakup + social_care_mh + homeless, data = pheRed)

Residuals:
    Min      1Q  Median      3Q     Max
-4.2494 -1.1217 -0.1575  0.9558  4.1356

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)           4.8392785  1.4371175   3.367 0.000977 ***
children_leaving_care 0.0423124  0.0197194   2.146 0.033595 *
self_harm_persons     0.0056140  0.0022875   2.454 0.015329 *
opiates               0.1055031  0.0507785   2.078 0.039537 *
marital_breakup       0.1878726  0.1344548   1.397 0.164505
social_care_mh        0.0008995  0.0004784   1.880 0.062108 .
homeless             -0.2066931  0.0749597  -2.757 0.006593 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.69 on 142 degrees of freedom
Multiple R-squared:    0.4, Adjusted R-squared:  0.3746
F-statistic: 15.78 on 6 and 142 DF,  p-value: 7.784e-14
```

```
autoplot(model)
```



Is this a good model? Maybe, but there appear to be outliers.

Now we are going to learn more about that bottom left plot.

11.3 Leverage and Influence

We have several observations in our dataset which are composed an observed value of y_i and the corresponding predictor variables $x_{1i}, x_{2i}, \ldots, x_{p_i}$.

We make a prediction

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_{1i} + \hat{\beta}_2 x_{2i} + \ldots + \hat{\beta}_p x_{pi}$$

Leverage is how much potential influence an observation has on a regression line.

Leverage for the i^{th} observation in a dataset is denoted by h_i.

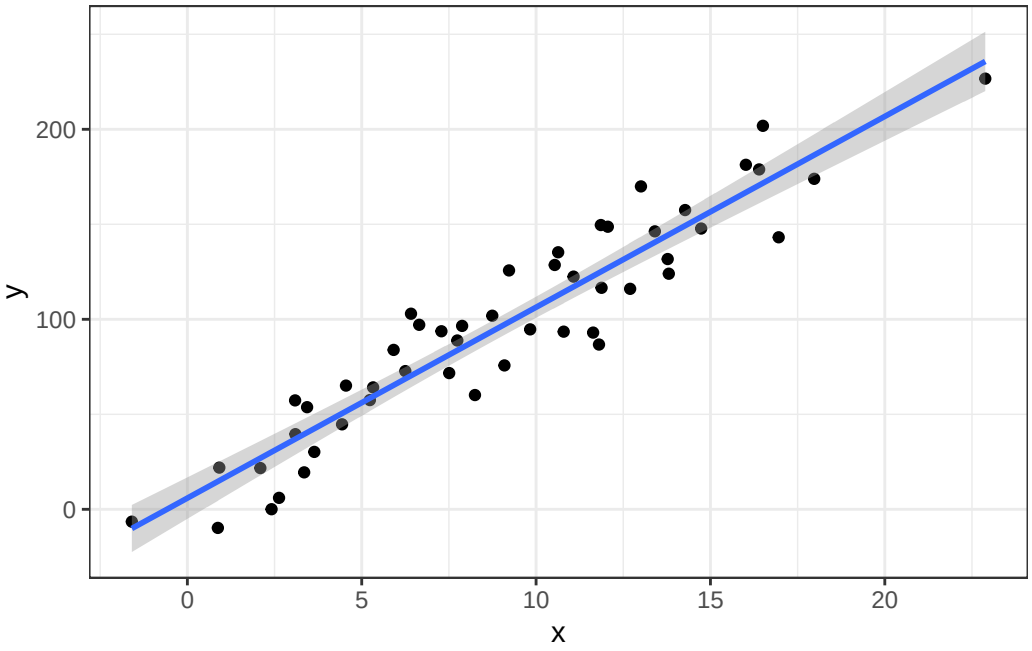
Leverage is a measure of how far $x_{1i}, x_{2i}, \ldots, x_{p_i}$ deviate from the rest of the predictor variable observations.

Influence is a measure of how much of an effect the i^{th} observation has on the regression line/surface.

11.3.1 Low/High leverage versus Low/High Influence

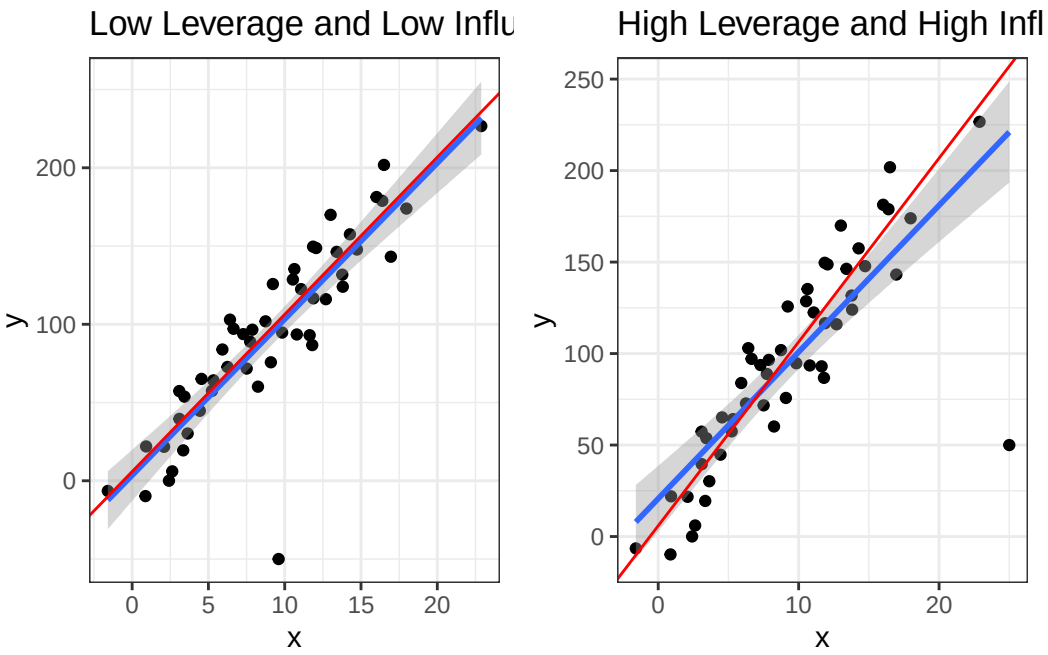
For simplicities sake, we'll look at this with just simple linear regression model.

Here's a regression model with perfectly well behaved



Now here are two plots. They each have an outlier. In red is the regression line that results from the outlier being removed.

- The left plot has an outlier that is close to the mean of x, and therefore has low leverage. Since the lines are close, this the outlier is low influence.
- The one one the right shows an outlier with high distance from the center of x equating to high leverage. The discrepancy between the lines with and without the outlier indicates high influence.



Coefficients for model with no outliers:

```
coeff.summary(lm(y~x, data))
```

term	estimate	std.error	statistic	p.value
(Intercept)	5.896215	5.4265974	1.08654	0.2826666
x	10.044226	0.5231077	19.20107	0.0000000

Coefficients for the the model with a high leverage but low influence outlier.

```
coeff.summary(lm(y~x, dat1))
```

term	estimate	std.error	statistic	p.value
(Intercept)	3.527027	8.1027541	0.4352875	0.6652652
x	9.975550	0.7821832	12.7534696	0.0000000

Coefficients for the the model with a high leverage but high influence outlier.

```
coeff.summary(lm(y~x, dat2))
```

term	estimate	std.error	statistic	p.value
(Intercept)	20.710068	8.9305015	2.319026	0.0246064
x	8.013461	0.8229571	9.737397	0.0000000

11.4 Finding High Influence Points

There are three main methods for determining high influence points:

- DFFITS: Determines effect of observation i on its estimate $\hat{y}_i$.
- Cook's Distance (Cook's D): Determines effect of an observation on the overall regression surface.
  - Cook's D and DFFITS are nearly identical accept for some slight tweaks. They have different cutoffs for "troublesome" values but tend to agree.
- DFBETAS: Determines effect of an observation on the each individual predictor coefficient.
  - Influence is determined by an observation being an outlier with respect to a predictor variable.
  - Some outliers only are outliers in terms of a single predictor variable.

11.4.1 DFFITS

- $\hat{y}_i$ is the predicted value of observation i in the data.
- $\hat{y}_{i(i)}$ is the predicted value of observation i in the data when the regression line is computed without observation i.
  - The notation kind of sucks, IMO, but it's difficult to communicate the information in such a compact form. Just repeat the definitions in your head 10 times.
- h_i is the leverage of observation i.
- $MSE_{(i)}$ is the MSE of the regression model without observation i.

If $\widehat{y_i}$ and $\widehat{y_{i(i)}}$ differ by a "substantial" relative to the leverage, then the outlier may be considered problematic

$$DFFITS_i = \frac{\hat{y}_i - \hat{y}_{i(i)}}{\sqrt{MSE_{(i)} \cdot h_i}}$$

There are a few different ways of determining if an observation has a "large" DFFITS value.

- For small to medium datasets, a $DFFITS$ exceeding 1 (or -1) is problematic.
- For large datasets a $DFFITS$ exceeding $2\sqrt{p/n}$. Personally, I'd recommend $3\sqrt{p/n}$ since DFFITS values are related to the t distribution.
  - Recall p is the number of predictors.
  - I'd say consider "large" to be 200 to 300 or more.
  - This stuff is from way back when getting "large" amounts of data was quite a bit harder/expensive.
  - Defining "large" is such an ephemeral thing given this age of "big" data.

11.4.2 Cook's Distance (D)

For obersvation i, Cook's Distance D_i is:

$$D_i = \frac{\sum_{i=1}^{n}(\hat{y}_i - \hat{y}_{i(i)})}{p \cdot MSE}$$

* Investigate observations with $D_i > 0.5$, though some suggest $D_i > 1$. * Cutoffs are a mess honestly, you shold look for D_i values that stick out and investigate.

11.4.3 DFBETAS

$DFBETA_{k(i)}$ is a measure of how much observation i affects the estimated coefficient $\hat{\beta}_k$

- $\hat{\beta}_k$ is the estimated coefficient using the whole dataset.
- $\hat{\beta}_{k(i)}$ estimated coefficient when observation i is removed from the data.
- $k = 0, 1, 2, ... p$
- $MSE_{(i)}$ is the MSE of the regression model without observation i.

$$DFBETA_{k(i)} = \frac{\hat{\beta}_k - \hat{\beta}_{k(i)}}{\sqrt{MSE_{(i)}c_{kk}}}$$

* A simple cutoff for $DFBETA_{k(i)} > 1$ indicates observation i has a large effect on coefficient k. * In this situation, you should care what happens to the y-intercept $\hat{\beta}_0$

- $c_{kk} is a value computed using matrix theory. We aren't a theory course so just trust me, it's a number that should be there. It will be caclulated for you.

11.4.4 Custom Functions: Influential Observations calculator

In R, you do not need to install packages if you know how to program your own function.

You can create a function that does what you want. Here is a function that calculates all of the influence measures for all the observations in your dataset.

Just run the code-chunk and now you can use the function.

```
influence.measures <- function (model){
    is.influential <- function(infmat, n) {
        k <- ncol(infmat) - 2
        if (n <= k)
            stop("too few cases, n < k")
        absmat <- abs(infmat)
        result <- cbind(absmat[, 1L:k] > 1, absmat[, k + 1] >
            1, infmat[, k + 2]> 0.5)
        dimnames(result) <- dimnames(infmat)
        result
    }
    infl <- influence(model)
    p <- model$rank
    e <- weighted.residuals(model)
    s <- sqrt(sum(e^2, na.rm = TRUE)/df.residual(model))
    mqr <- stats:::qr.lm(model)
    xxi <- chol2inv(mqr$qr, mqr$rank)
    si <- infl$sigma
    h <- infl$hat
    dfbetas <- infl$coefficients/outer(infl$sigma,
                                       sqrt(diag(xxi)))
    vn <- variable.names(model)
    vn[vn == "(Intercept)"] <- "1_"
    colnames(dfbetas) <- paste("dfb", vn, sep = ".")
    dffits <- e * sqrt(h)/(si * (1 - h))
    if (any(ii <- is.infinite(dffits)))
        dffits[ii] <- NaN
    cooks.d <- (if (inherits(model, "glm"))
        (infl$pear.res/(1 - h))^2 * h/(summary(model)$dispersion *
            p)
    else ((e/(s * (1 - h)))^2 * h)/p)
    infmat <- cbind(dfbetas, dffit = dffits,
        cook.d = cooks.d)
    infmat[is.infinite(infmat)] <- NaN
```

```
    is.inf <- is.influential(infmat, sum(h > 0))
    infmat %>%
      as.data.frame() %>%
      mutate(influential = apply(is.inf, 1, any))
}
```

11.4.5 Influence Measures on PHE data

`TRUE` or `FALSE` is indicated in the right most column `influential` for observations that are declared "influential" accoding to the cutoffs discussed.

```
check <- influence.measures(model)

### This filters out any observations that are marked as "influential"

filter(check, influential)
```

```
       dfb.1_ dfb.children_leaving_care dfb.self_harm_persons dfb.opiates
26 -0.2156744                0.19024198            0.13998317  -0.5959013
78  0.1388500                0.08110425            0.07286781   0.7102005
   dfb.marital_breakup dfb.social_care_mh dfb.homeless    dffit    cook.d
26          0.06112809           1.908796   0.05305668 1.987504 0.5367057
78         -0.39154530           0.405517  -0.21624790 1.151952 0.1821697
   influential
26        TRUE
78        TRUE
```

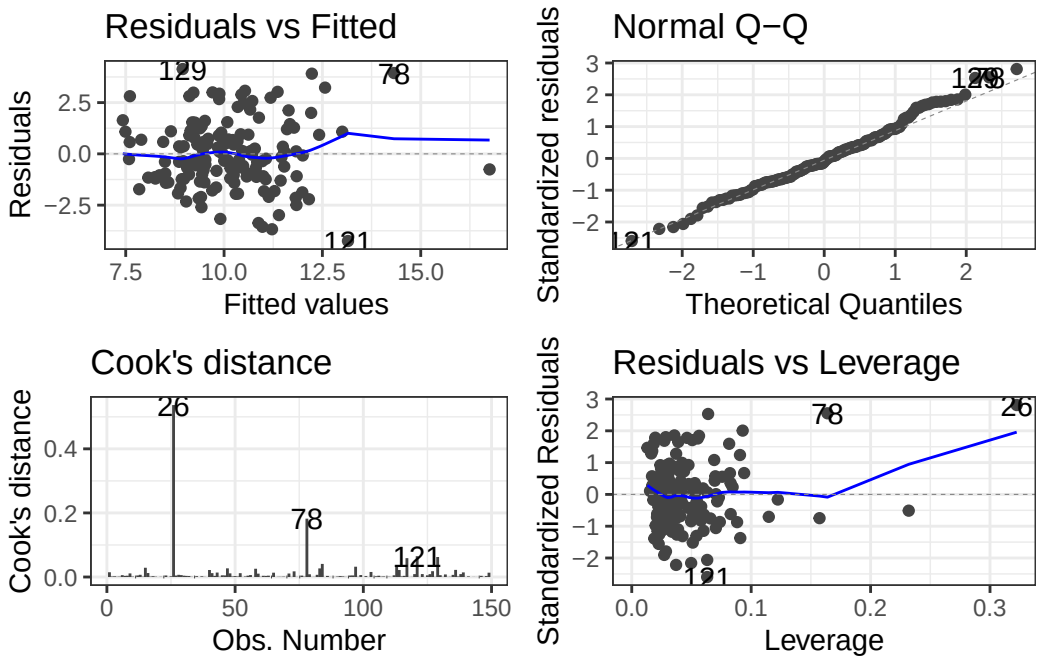
11.4.6 Plotting the Residuals, Cook's Distance and Leverage

- The `autoplot` function has a `which` argument.
- It can create a total of 6 plots, each one having to do with residuals and influence measures.
- `autoplot(model, which = c(1, 2, 3, 4, 5, 6))` will plot all 6.
  - Plot 1 is Residuals vs Fitted
  - Plot 2 is the Normal Q-Q plot
  - Plot 3 is the scale-location plot,
  - Plot 4 is a plot of Cook's distance for each observation.
  - Plot 5 is Residuals vs Leverage
  - Plot 6 is Cook's Distance vs Leverage

- Remove any numbers for plots you don't want.

My personal preference would be 1, 2, 4, 5.

```
autoplot(model, which = c(1, 2, 4, 5))
```



- The top left is the raw residuals, this is where we assess the bias and constant variance.
- The bottom right, are residuals calculated based on leverage.
  - If the Standardized/Studentized Residual gets close to a value of 3, it may be problematic.

Studentized Residuals (Sometimes called Standardized Residuals):

$$t_i = \frac{e_i}{\sqrt{MSE_{(i)}(1-h_i)}}$$

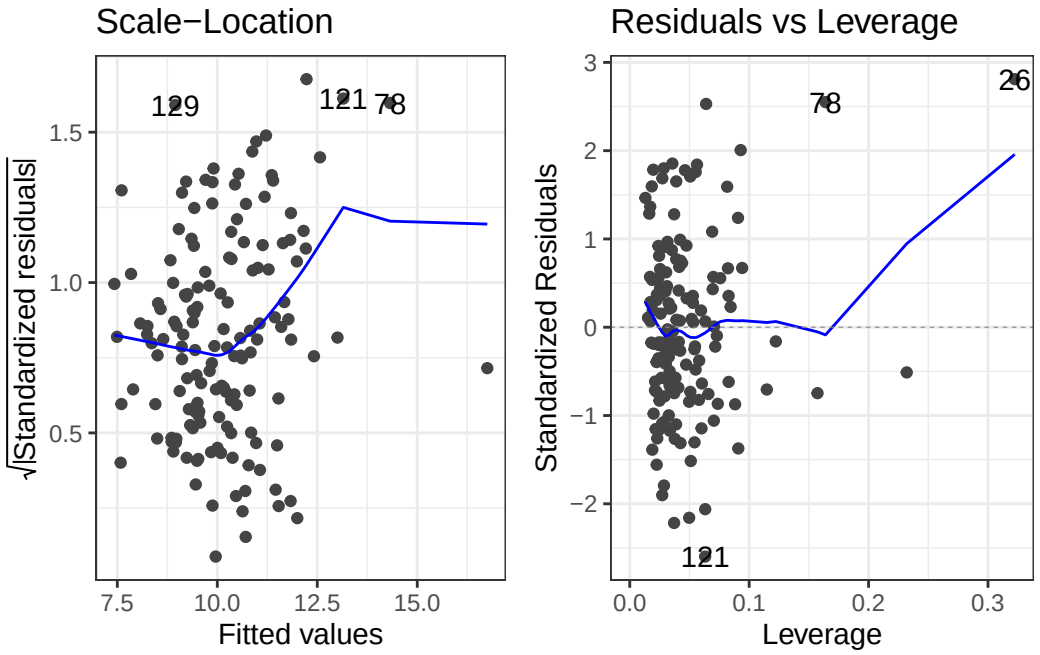
It is EXTREMELY frustrating the language used.

Sometimes Standardized Residuals are:

$$t_i = \frac{e_i}{\sqrt{MSE}}$$

Which *doesn't* account for leverage.

```
autoplot(model, which = c(3,5))
```



If both plots used the same definition of standard residuals, the marked observations which are the three most extreme residuals should be the observations.

:eyeroll:

11.5 You found some values that are high influence outliers, now what?

If there are only a couple per 200 or so, you can *probably* just delete them and not worry about it. If you have several, then there might be a bigger issue.

Ideally, you have more intimate knowledge of the data and would identify why outliers are not representative of the general population you are trying to model. If so, deletion probably is just fine.

Anyway, let's pretend that that we can delete the two observations.

Let's start with the worst one, 26.

11.5.1 Removing 26

```
pheOut1 <- pheRed[-26, ]

modelOut1 <- lm(suicide_rate ~ children_leaving_care + self_harm_persons +
    opiates + marital_breakup + social_care_mh + homeless, pheOut1)

summary(model)
```

```
Call:
lm(formula = suicide_rate ~ children_leaving_care + self_harm_persons +
    opiates + marital_breakup + social_care_mh + homeless, data = pheRed)

Residuals:
    Min      1Q  Median      3Q     Max
-4.2494 -1.1217 -0.1575  0.9558  4.1356

Coefficients:
                        Estimate Std. Error t value Pr(>|t|)
(Intercept)            4.8392785  1.4371175   3.367 0.000977 ***
children_leaving_care  0.0423124  0.0197194   2.146 0.033595 *
self_harm_persons      0.0056140  0.0022875   2.454 0.015329 *
opiates                0.1055031  0.0507785   2.078 0.039537 *
marital_breakup        0.1878726  0.1344548   1.397 0.164505
social_care_mh         0.0008995  0.0004784   1.880 0.062108 .
homeless              -0.2066931  0.0749597  -2.757 0.006593 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.69 on 142 degrees of freedom
Multiple R-squared:    0.4, Adjusted R-squared:  0.3746
F-statistic: 15.78 on 6 and 142 DF,  p-value: 7.784e-14
```

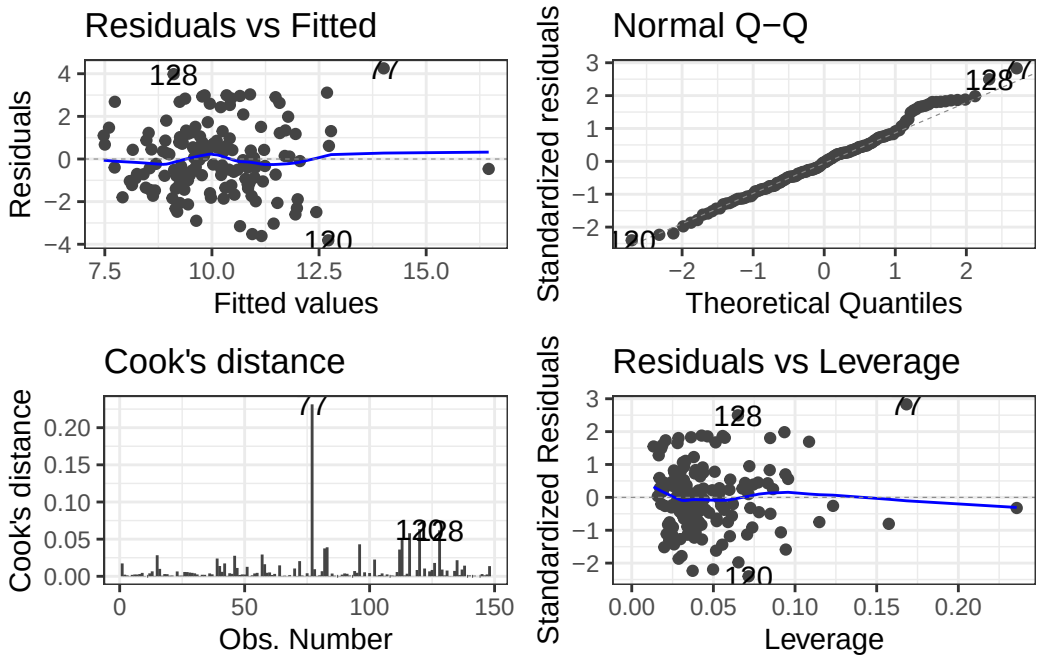
```
summary(modelOut1)
```

```
Call:
lm(formula = suicide_rate ~ children_leaving_care + self_harm_persons +
    opiates + marital_breakup + social_care_mh + homeless, data = pheOut1)
```

```
Residuals:
    Min      1Q  Median      3Q     Max
-3.8140 -1.0888 -0.0478  0.8991  4.2533

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)            5.142e+00  1.405e+00   3.658 0.000358 ***
children_leaving_care  3.865e-02  1.927e-02   2.006 0.046811 *
self_harm_persons      5.302e-03  2.233e-03   2.374 0.018957 *
opiates                1.350e-01  5.057e-02   2.670 0.008479 **
marital_breakup        1.799e-01  1.312e-01   1.371 0.172448
social_care_mh         9.008e-06  5.596e-04   0.016 0.987180
homeless              -2.106e-01  7.312e-02  -2.880 0.004598 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.648 on 141 degrees of freedom
Multiple R-squared:  0.4011,    Adjusted R-squared:  0.3756
F-statistic: 15.74 on 6 and 141 DF,  p-value: 8.76e-14
```

The biggest change is in the `social_care_mh` variable. It seems almost complete useless now.

```
autoplot(modelOut1, which = c(1, 2, 4, 5))
```



11.5.2 Does stepwise

What if we did stepwise regression?

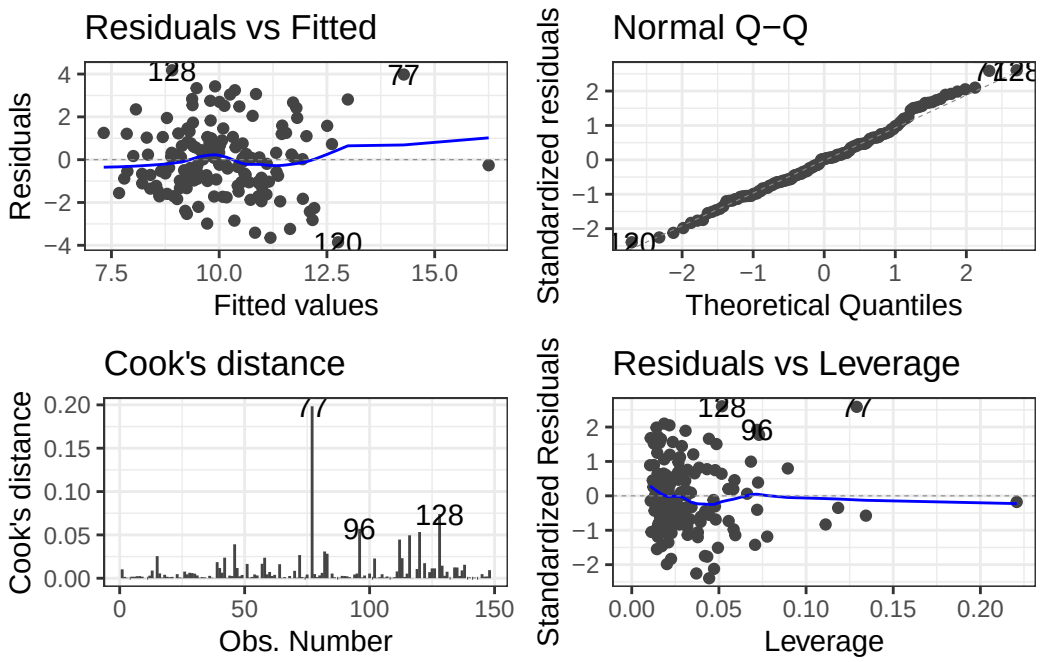
```
fullModelOut1 <- lm(suicide_rate ~ ., pheOut1)
stepModelOut1 <- step(fullModelOut1,
                      direction = "both", trace = 0)

summary(stepModelOut1)
```

```
Call:
lm(formula = suicide_rate ~ children_leaving_care + self_harm_persons +
    opiates + homeless, data = pheOut1)

Residuals:
    Min      1Q  Median      3Q     Max
-3.8571 -1.0684  0.0412  1.0111  4.1770

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)           6.936358   0.526026  13.186  < 2e-16 ***
```

```
children_leaving_care  0.044349   0.018584   2.386  0.01832 *
self_harm_persons      0.006252   0.002120   2.948  0.00373 **
opiates                0.133822   0.048731   2.746  0.00681 **
homeless              -0.222190   0.072526  -3.064  0.00261 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.647 on 143 degrees of freedom
Multiple R-squared:  0.393, Adjusted R-squared:  0.376
F-statistic: 23.15 on 4 and 143 DF,  p-value: 9.143e-15
```

Now stepwise regression says the best model (according to AIC) appears to now only have four variables with `marital_breakup` and `social_care_mh` now out of the model.

```
autoplot(stepModelOut1, which = c(1, 2, 4, 5))
```



11.5.3 Removing the other outlier

The other outlier in the original data was 78, which is now 77 in the data with the first outlier removed.

```r
pheOut2 <- pheRed[-c(26, 78), ]
modelOut2 <- lm(suicide_rate ~ children_leaving_care + self_harm_persons +
    opiates + marital_breakup + social_care_mh + homeless, pheOut2)

summary(modelOut2)
```

```
Call:
lm(formula = suicide_rate ~ children_leaving_care + self_harm_persons +
    opiates + marital_breakup + social_care_mh + homeless, data = pheOut2)

Residuals:
    Min      1Q  Median      3Q     Max
-3.6411 -1.1723 -0.0252  0.9400  3.7675

Coefficients:
                        Estimate Std. Error t value Pr(>|t|)
(Intercept)            4.9507489  1.3713937   3.610 0.000426 ***
children_leaving_care  0.0366938  0.0187964   1.952 0.052914 .
self_harm_persons      0.0051023  0.0021779   2.343 0.020552 *
opiates                0.0987500  0.0508446   1.942 0.054121 .
marital_breakup        0.2352779  0.1292469   1.820 0.070838 .
social_care_mh        -0.0002604  0.0005533  -0.471 0.638623
homeless              -0.1936071  0.0715014  -2.708 0.007619 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.606 on 140 degrees of freedom
Multiple R-squared:  0.3686,    Adjusted R-squared:  0.3416
F-statistic: 13.62 on 6 and 140 DF,  p-value: 3.809e-12
```

What does stepwise regression say now?

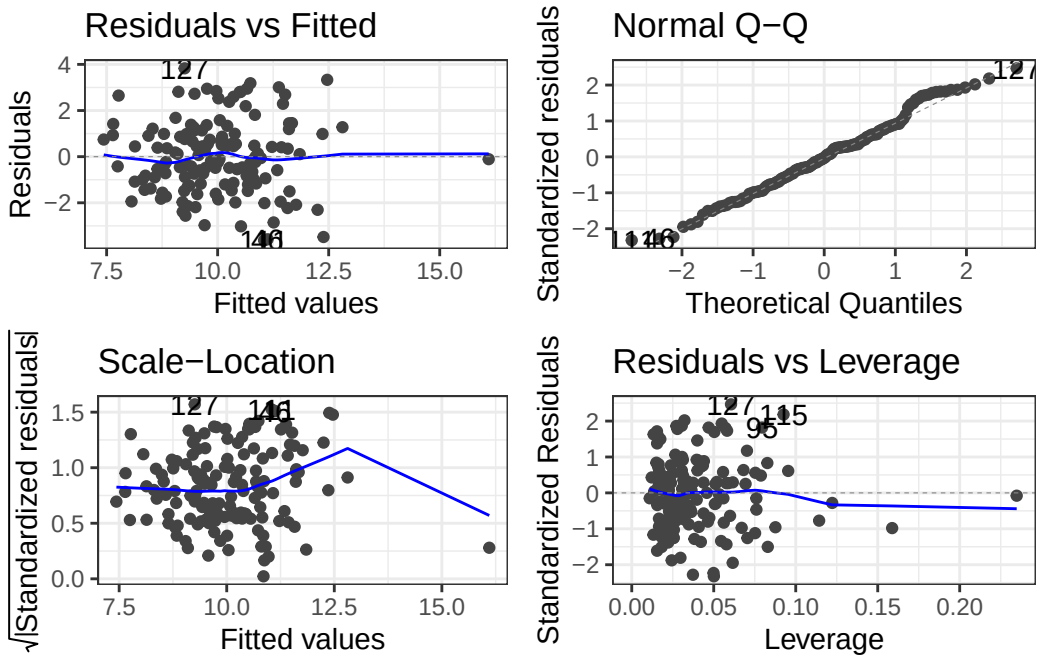
```r
fullModelOut2 <- lm(suicide_rate ~ ., pheOut2)
stepModelOut2 <- step(fullModelOut2, direction = "both", trace = 0)

summary(stepModelOut2)
```

```
Call:
lm(formula = suicide_rate ~ children_leaving_care + self_harm_persons +
```

```
    opiates + marital_breakup + homeless, data = pheOut2)

Residuals:
    Min      1Q  Median      3Q     Max
-3.6265 -1.1113 -0.0436  0.9571  3.8287

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)            4.926482   1.366636   3.605 0.000432 ***
children_leaving_care  0.038065   0.018518   2.056 0.041667 *
self_harm_persons      0.005157   0.002169   2.378 0.018764 *
opiates                0.093673   0.049550   1.890 0.060745 .
marital_breakup        0.228541   0.128097   1.784 0.076553 .
homeless              -0.192514   0.071266  -2.701 0.007753 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.602 on 141 degrees of freedom
Multiple R-squared:  0.3676,    Adjusted R-squared:  0.3452
F-statistic:  16.4 on 5 and 141 DF,  p-value: 9.804e-13
```

And now `marital_breakup` breakup is back in?

```
autoplot(stepModelOut2)
```



11.6 Which model to use

In short, there is no good answer.

- If you have absolutely no idea what's going on and all you care about is prediction accuracy, the stepwise regression approach *may* (read: MAY) be okay.
  - This is an acceptable approach within exams and assignments for this course, but should be scrutinized in real world modeling.
- If you know more about what's going on and have specific experimental questions, then use the models proposed by those questions and look at how useful they are.
  - That's kind of what they did in the paper.

There is no correct way to go about this.

11.7 Our model building process

1. Look at the full model. Coefficients, residuals, and all.
2. Investigate for multicollinearity and find ways to remove variables that may be problematic.

3. Look at the full model of the reduced data: coefficients, residuals, and all. You may need to apply a transformation to the y variable.
4. If you think it's a good fit and can justify using it, you're done.
5. Otherwise you need to start eliminating variables via knowledge of the data or specific experimental questions ideally. If you are purely aiming for a predictive approach, you may try stepwise regression. It is best to try all methods, but stepwise in both directions is acceptable in this class.
6. Once you have found a reduced model, examine the residuals for outliers and violation assumptions. Remove outliers if you can justify it.
7. If you don't have outliers and your assumptions look good, you're done.
8. Check if your variables still remain relevant. You may have to remove or add variables that are now relevant. Use experimental questions or stepwise regression again…
9. Check residuals and outliers again. Hopefully everything looks. Good.

Everytime you messed with something in the model, you need to go back through and check residuals, validity of transformations, etc.

I'll throw you softballs in this class for the sake of your sanity.

12 One-Way ANOVA

Here are some code chunks that setup this chapter.

```
# Here are the libraries I used
library(tidyverse) # standard
library(knitr) # need for a couple things to make knitted document to look nice
library(readr) # need to read in data
library(ggpubr) # allows for stat_cor in ggplots
library(ggfortify) # Needed for autoplot to work on lm()
library(gridExtra) # allows me to organize the graphs in a grid
library(car) # need for some regression stuff like vif
library(GGally)
```

```
# This changes the default theme of ggplot
old.theme <- theme_get()
theme_set(theme_bw())
```

Comparing More Than Two Group Means: Analysis of Variance (ANOVA or AOV)

12.1 Review: Comparing Two Groups (Sections 7.1 - 7.7 of JB Statistics)

Two-Sample Tests

$H_0 : \mu_1 = \mu_2$

$H_1 : \mu_1 \neq \mu_2$

12.1.1 The two-sample t-test: Pooled

Student's two-sample t-test, and assumes unknown but equal population variances/standard deviations, i.e.,$\sigma_1 = \sigma_2$.

We use a **pooled sample variance** estimate:

$$s_p^2 = \frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{(n_1 - 1) + (n_2 - 1)}$$

$$t = \frac{\bar{y_1} - \bar{y_2}}{\sqrt{s_P^2 \left(\frac{1}{n_1} + \frac{1}{n_2}\right)}}$$

And the degrees of freedom is $df = n_1 + n_2 - 2$

- Pro: Powerful if $\sigma_1 = \sigma_2$.
- Con: When is that true?

12.1.2 Welch's two-sample t-test

Assume $\sigma_1 \neq \sigma_2$

$$t = \frac{\bar{y_1} - \bar{y_2}}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}}$$

$$df = \frac{\left(\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}\right)^2}{\frac{\left(s_1^2/n_1\right)^2}{n_1 - 1} + \frac{\left(s_2^2/n_2\right)^2}{n_2 - 1}}$$

12.1.3 R command, `t.test()`

```
t.test(x, y)
```

By default, this performs Welch's test. If you must perform Student's test:

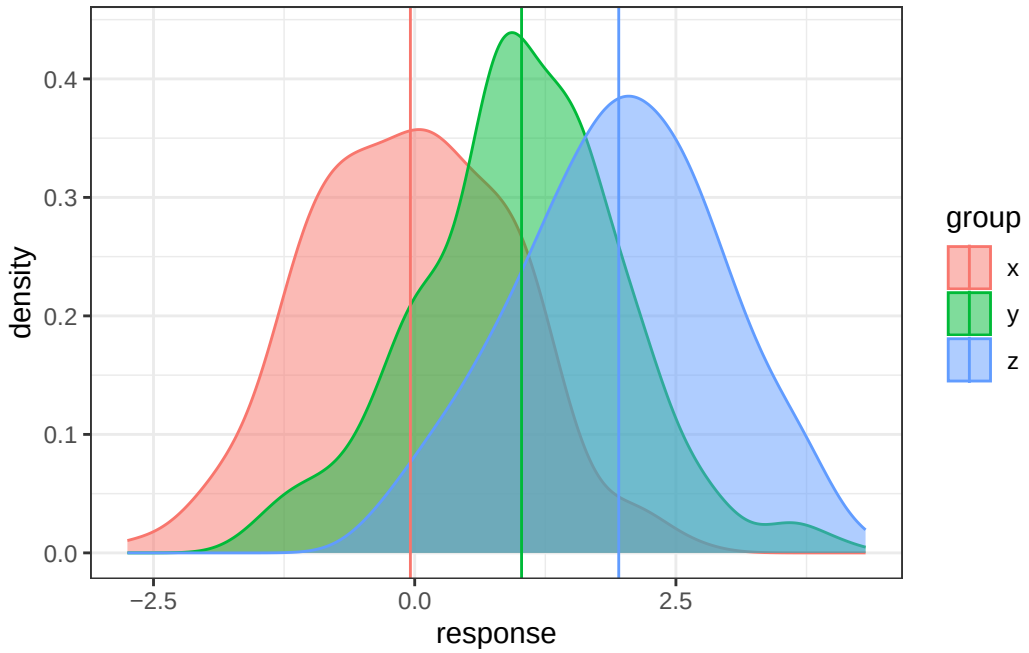
```
t.test(x, y, var.equal=TRUE)
```

12.1.4 Hypothetical Example: Three Groups

Let's consider having three groups.

- In the code below, the groups are generated and technically we know the populations means.
- But we will only have the sample data in reality.
- How can we use three sample means at once to distinguish between three groups?

```
n = 200

data <- data.frame(x = rnorm(n, 0, 1),
                   y = rnorm(n, 1, 1), z = rnorm(n, 2, 1))
data <- gather(data, key = 'group', value = 'response')

# sample means

sample.means <- aggregate(x = response ~ group ,
                          data = data, FUN = mean)
sample.means$mu = c(0,1,2)

ggplot(data, aes(x = response, y = after_stat(density),
                 color = group, fill = group)) +
  geom_density(alpha = 0.5) +
  geom_vline(data = sample.means, aes(xintercept = response,
                                      color = group))
```



There is sampling variability associated with each sample mean. Run this code a few times do see how much the sample means will change from sample to sample. How do we account for the sampling variability when comparing three groups simultaneously? How do we even compare three groups simultaneously?

12.2 Analysis of Variance

12.2.1 General Objective of Analysis of Variance (ANOVA)

First, we are going to have a hypothesis test involved with comparing several groups.

- The null hypothesis is going to be the similar to before, all the groups have equivalent population means.
- The alternative is a bit trickier...

Hypotheses:

$H_0 : \mu_1 = \mu_2 = \cdots = \mu_t$
$H_1 :$ not that...

Why not just do a bunch of separate t-tests?

If we have three groups:

- Compare group 1 to group 2.
- Compare group 1 to group 3.
- Compare group 2 to group 3.

And that would account for all possible pairings of groups. However, each of these would be a separate hypothesis test.

These are called **pair-wise comparisons**.

In general, if there are t groups, there are

$$\binom{t}{2} = \frac{t(t-1)}{2}$$

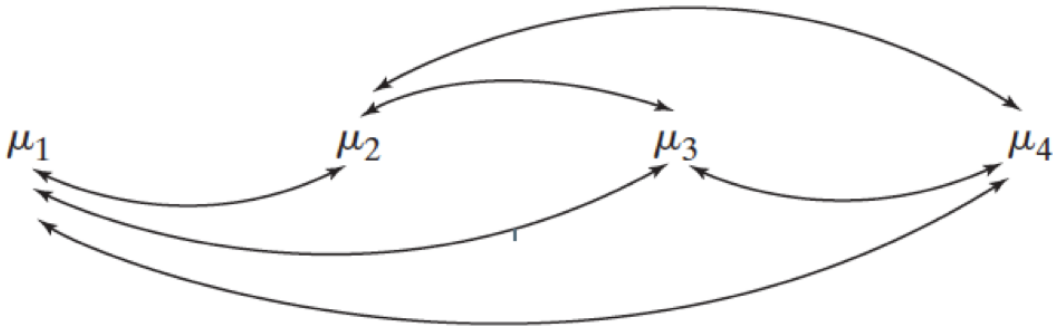
unique pair-wise comparisons.

12.2.2 Familywise Error Rate: What happens when you do multiple hypothesis tests

Say we had four treatment groups and we wanted to compare the means of all four groups to see if any differed.

Through pairwise comparisons, we would end up with $\frac{4 \cdot 3}{2} = 6$ total comparisons.

$$H_0 : \mu_1 = \mu_2 \quad H_0 : \mu_1 = \mu_3 \quad H_0 : \mu_1 = \mu_4$$
$$H_0 : \mu_2 = \mu_3 \quad H_0 : \mu_2 = \mu_4 \quad H_0 : \mu_3 = \mu_4$$

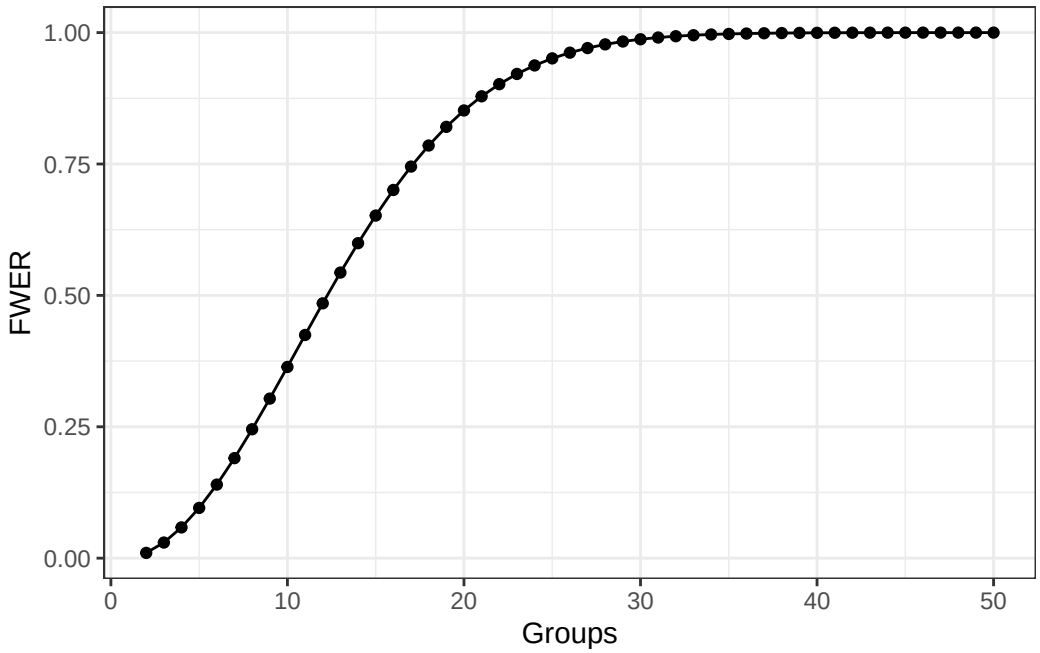


This would amount to 6 hypothesis tests to decide which if any group means differ.

- Assume each hypothesis test is done with the same Type I Error Rate α.
  - Pr(Reject H_0 | H_0 True)
- This is known as a **family** of hypothesis tests, a set of hypothesis tests under one objective.
- The **Family Wise Error Rate (FWER)** is the probability of making at least one Type I Error among a family of hypothesis tests.

$$\begin{aligned} FWER &= 1 - (1 - \alpha)^c \\ &> \alpha \text{ for } c = 2, 3, ... \end{aligned}$$

The following graph shows what happens to the FWER when the number of groups increases and each pairwise comparison done with $\alpha = 0.01$.


1.00
0.75
0.50
0.25
0.00
FWER
0
10
20
30
40
50
Groups

12.2.3 How ANOVA Works

The analysis of variance (ANOVA) is just that. It looks at two types of variability in the data.

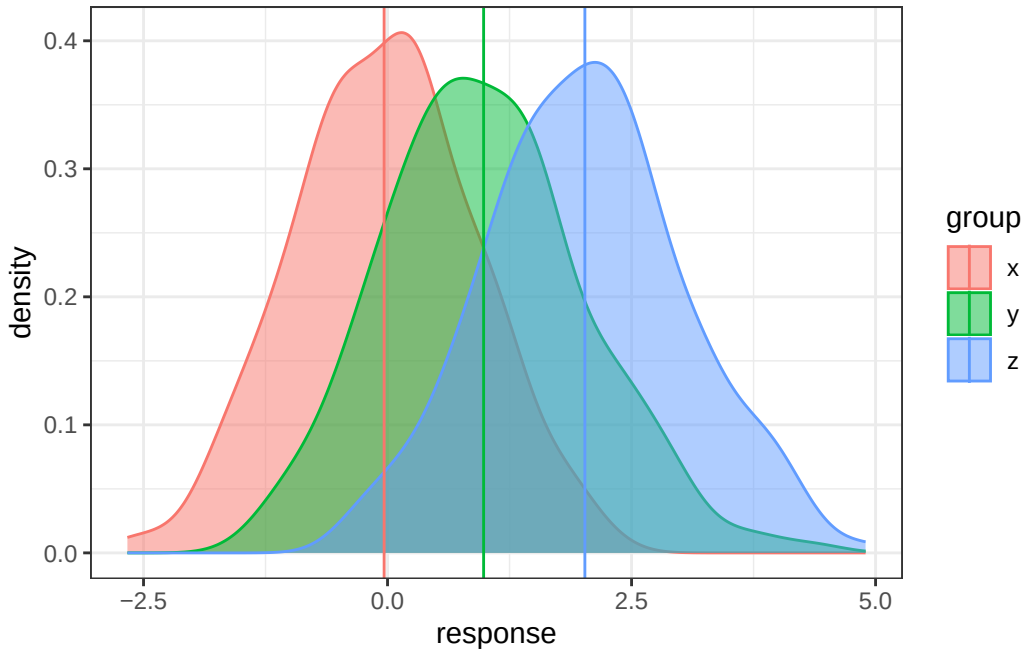
- **Between** or **Treatment** Variability: This is the variability *between* the groups which is measured by essentially computing a the variance of the sample means between the different groups.
- The **Within** or **Error** Variability: This is a collective measure of the variability within the groups.

If the Between/Treatment Variability is noticeably larger than the the Within/Error variability, we have strong evidence in favor that the least some groups have different population means.

12.2.4 Treatment versus Error Variability Demos

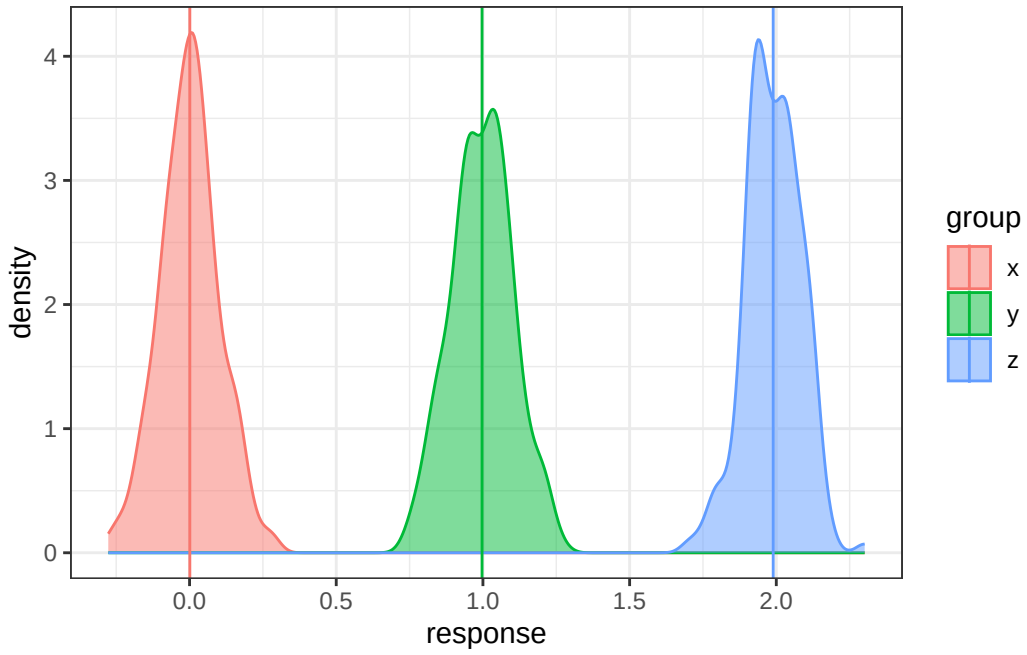
Here is a simulated dataset with moderately small within/error variability relative to the between/treatment variability

```
n = 200

data <- data.frame(x = rnorm(n, 0, 1),
                   y = rnorm(n, 1, 1), z = rnorm(n, 2, 1))
data <- gather(data, key = 'group', value = 'response')

sample.means <- aggregate(x = response ~ group ,
                          data = data, FUN = mean)
sample.means$mu = c(0,1,2)

ggplot(data, aes(x = response, y = after_stat(density),
                 color = group, fill = group)) +
  geom_density(alpha = 0.5) +
  geom_vline(data = sample.means, aes(xintercept = response,
                                      color = group))
```


density
0.4
0.3
0.2
0.1
0.0
−2.5
0.0
2.5
5.0
response
group
x
y
z

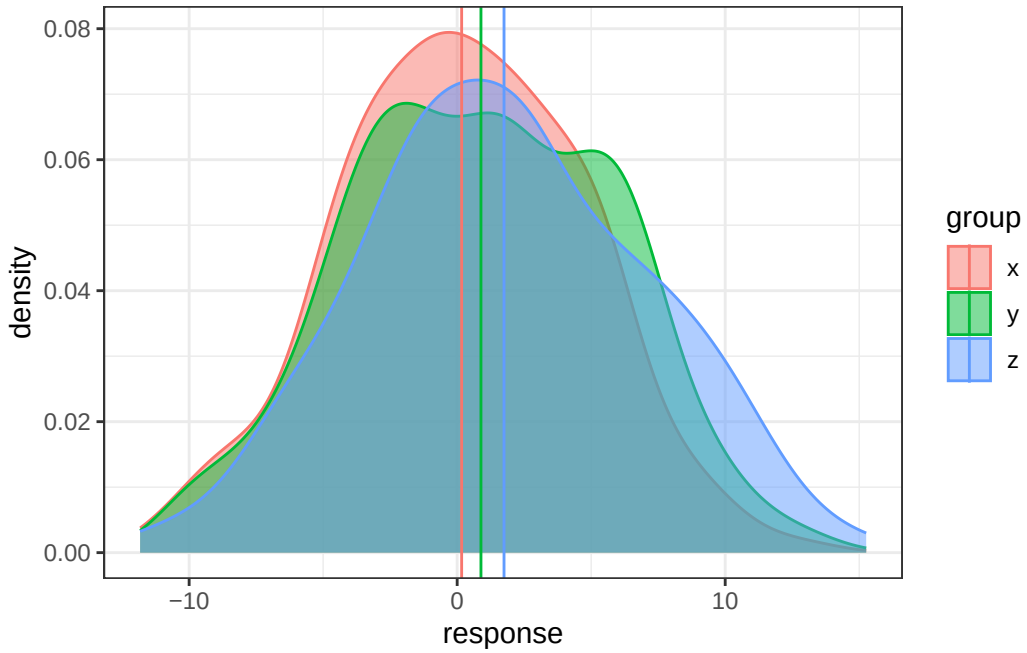
Here is an example a very small within/error variability relative to the between/treatment variability.

```
n = 200

data <- data.frame(x = rnorm(n, 0, 0.1),
                   y = rnorm(n, 1, 0.1), z = rnorm(n, 2, 0.1))
data <- gather(data, key = 'group', value = 'response')

# sample means

sample.means <- aggregate(x = response ~ group ,
                          data = data, FUN = mean)
sample.means$mu = c(0,1,2)

ggplot(data, aes(x = response, y = ..density..,
                 color = group, fill = group)) +
  geom_density(alpha = 0.5) +
  geom_vline(data = sample.means, aes(xintercept = response,
                                      color = group))
```



Lastly, large within/error variability relative to between/treatment variability.

```
n = 200

data <- data.frame(x = rnorm(n, 0, 5),
                   y = rnorm(n, 1, 5), z = rnorm(n, 2, 5))
data <- gather(data, key = 'group', value = 'response')

# sample means

sample.means <- aggregate(x = response ~ group ,
                          data = data, FUN = mean)
sample.means$mu = c(0,1,2)

ggplot(data, aes(x = response, y = ..density..,
                 color = group, fill = group)) +
  geom_density(alpha = 0.5) +
  geom_vline(data = sample.means, aes(xintercept = response,
                                      color = group))
```



- In each situation, the population means are the same: 0, 1, and 2.
- In some situations it's much easier to distinguish the groups than in others.

12.3 Formulating ANOVA: Notation

- y_{ij} is the j^{th} observation in the i^{th} treatment group.
  - $i = 1, \ldots, t$, where t is the total number of treatment groups.
  - $j = 1, \ldots, n_i$ where n_i is the number of observations in treatment group i.
- $\overline{y}_{i.} = \sum_{j=1}^{n_i} \frac{y_i j}{n_i}$ is the mean of treatment group i.
- $\bar{y}_{..} = \sum_{i=1}^{t} \sum_{j=1}^{n_i} \frac{y_{ij}}{N}$ is the over all mean of all observations.
  - N is the total number of observations.

12.3.1 Sums of Squares

Sum of Squares for the treatments

$$SST = \sum_{i=1}^{t} n_i \left(\overline{y}_{i.} - \bar{y}_{..}\right)^2$$

Sum of squares for the errors

$$SSE = \sum_{i=1}^{t} \sum_{j=1}^{n_i} \left(y_{ij} - \bar{y}_{i.}\right)^2$$

With each sum of squares, there is an associated degrees of freedom.

- Treatment: $df_t = t - 1$
- Error: $df_e = N - t$

12.3.2 Mean Squares

$$MST = \frac{SST}{t-1}$$

$$MSE = \frac{SSE}{N-t}$$

12.3.3 Test Statistic

$$F_t = \frac{MST}{MSE}.$$

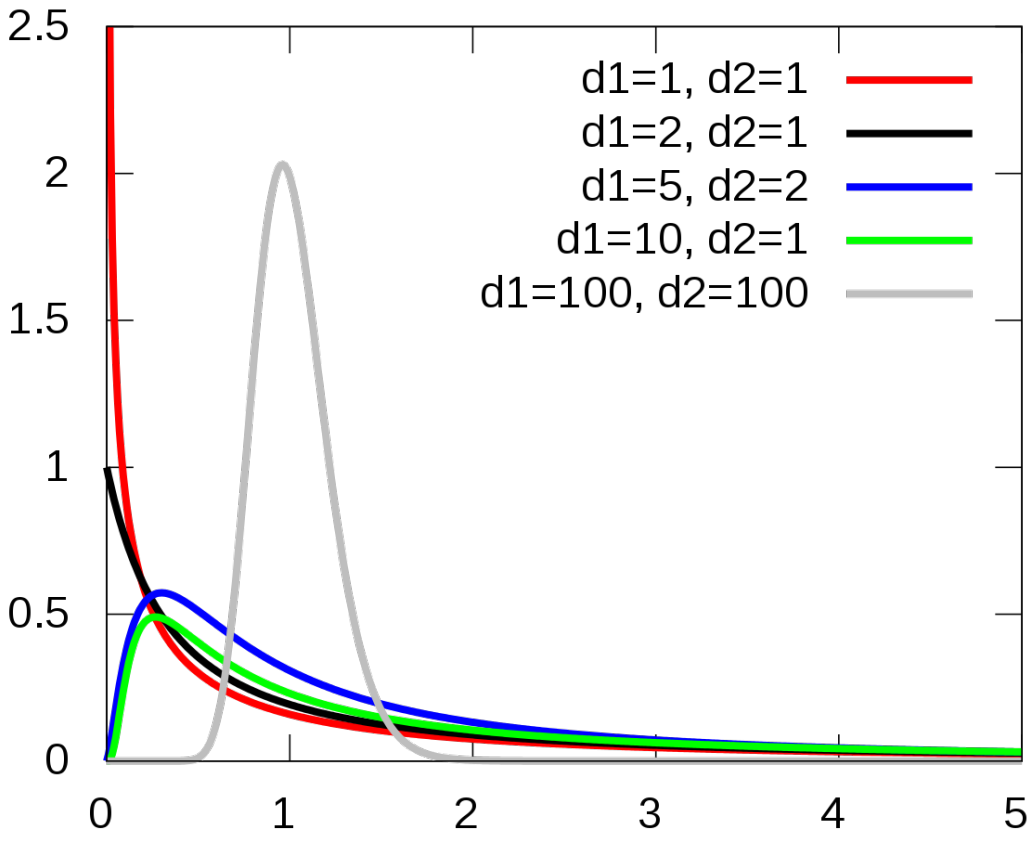
12.3.4 F-Distribution

Under the the following assumptions:

1. The null hypothesis is true, i.e., all groups have equivalent means,
2. The groups are indendent and normally distributed,
3. The groups all have equivalent variances,

the test statistic F_t follows an $F(t-1, N-t)$ distribution.

12.3.5 F-Distribution Visualization



12.4 How is this a "Linear Model"

12.4.1 Means Model

Means Model:

$y = \mu_i + \epsilon$

Which leads to the hypotheses in ANOVA:

$H_0 : \mu_1 = \cdots = \mu_t$

H_1: The means are not all equal.

12.4.2 Effects Model

Effects Model:

$y = \mu + \tau_i + \epsilon$

The τ_i's are the shift parameters that cause the groups to differ.

Which leads to an equivalent set of hypotheses:

$H_0 : \tau_1 = \cdots = \tau_t = 0$

$H_1 : \text{Not } H_0$

12.5 OASIS MRIs

Variables in OASIS data

- Group: whether subject is demented, nondemented, or converted.
- Visit: (Not relevant) which visit number of the MRI
- Gender: M or F
- MR Delay: (Not relevant) time between each successive MRI. First MRIs
- Hand: Handedness of subject. All patients were R (right-handed)
- Age: Age in years
- Educ: Years of education
- SES: Socioeconomic status 1 - 5, low to high SES (arbitrary cutoffs most likely)
- MMSE: Mini Mental State Examination
- CDR: Clinical Dementia Rating, 0 - 2. 0 is non-dementia, CDR $>$ 0 is severity of dementia.
- eTIV: Estimated Total Intracranial Volume (milliliters?)

- nWBV: Normalized Whole Brain Volume; Brain volume is normalized by intercranial volume to put subjects of different sizes and gender on the same scale. To my best knowledge…
- ASF: Atlas Scaling Factor; I tried investigating this but that's some rabbit hole that is very deep apparently. It has something to do with the normalization I think.

Unlike before, we're going to compare all three patient groups in the OASIS data: Demented, Nondemented, and Converted.|

Converted patients are those that entered the study that were no diagnosed with dementia, and then by the time the study ended they had been diagnosed with dementia.

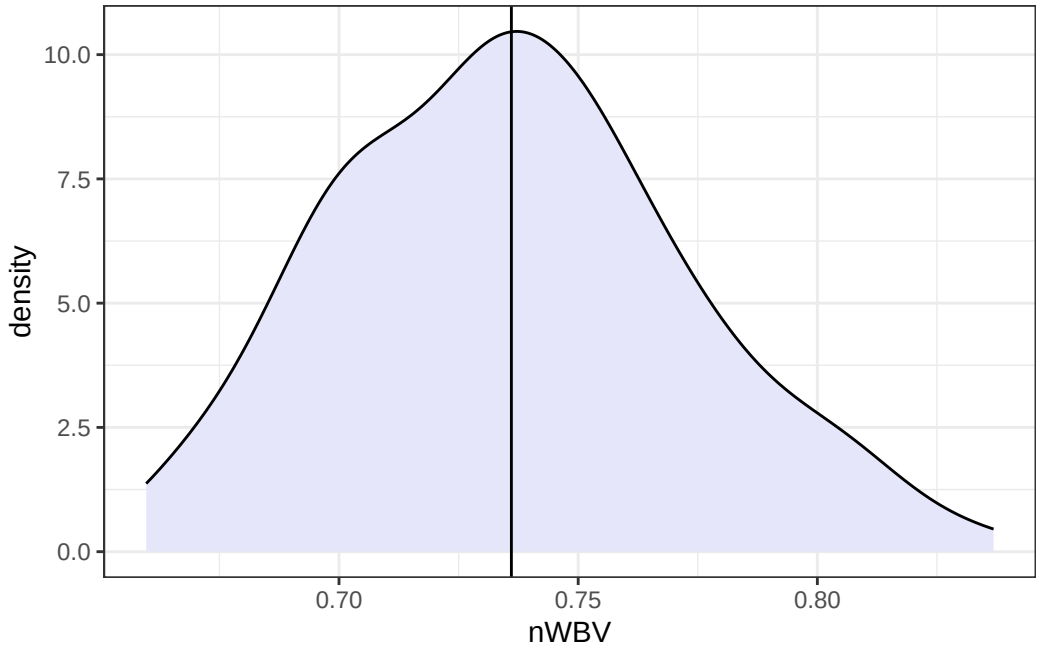
The objective is to assess the Normalized Whole Brain Volume `nWBV` of the different groups and see if the mean `nWBV` differs.

```
oasis <- read_csv(here::here("datasets",'oasis.csv'))
```

12.5.1 Examining the data

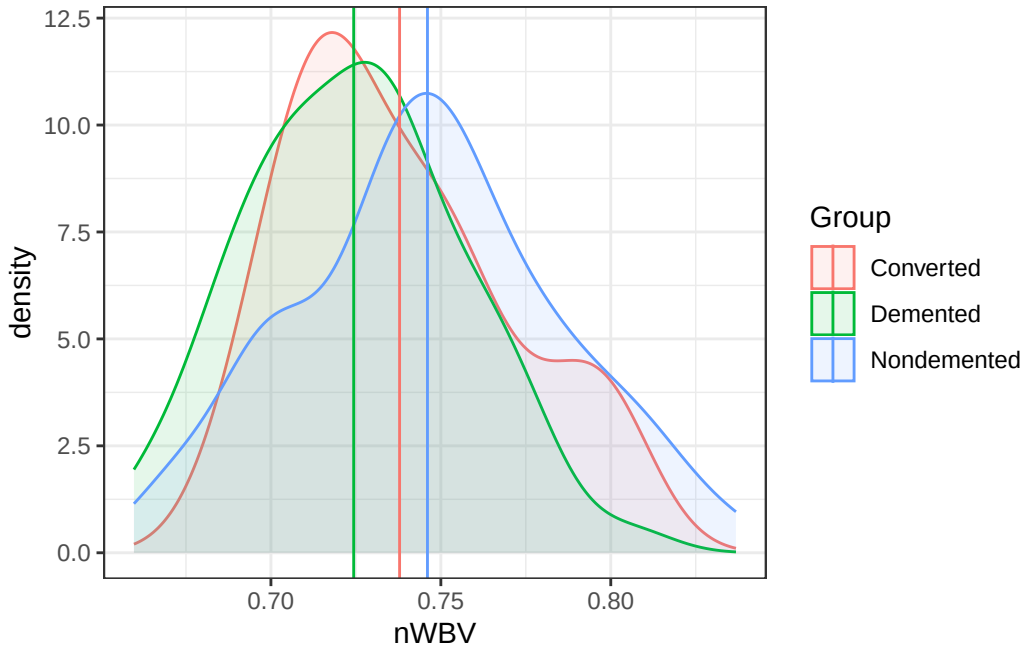
Let's look at the nWBVs overall.

```
nWBVmean <- mean(oasis$nWBV)

### geom_density() produces a "smoothed" histogram.
### if you wanted a histogram, you could use geom_histogram()


ggplot(oasis, aes(x = nWBV)) +
  geom_density(fill = "lavender") +
  geom_vline(xintercept = nWBVmean)
```

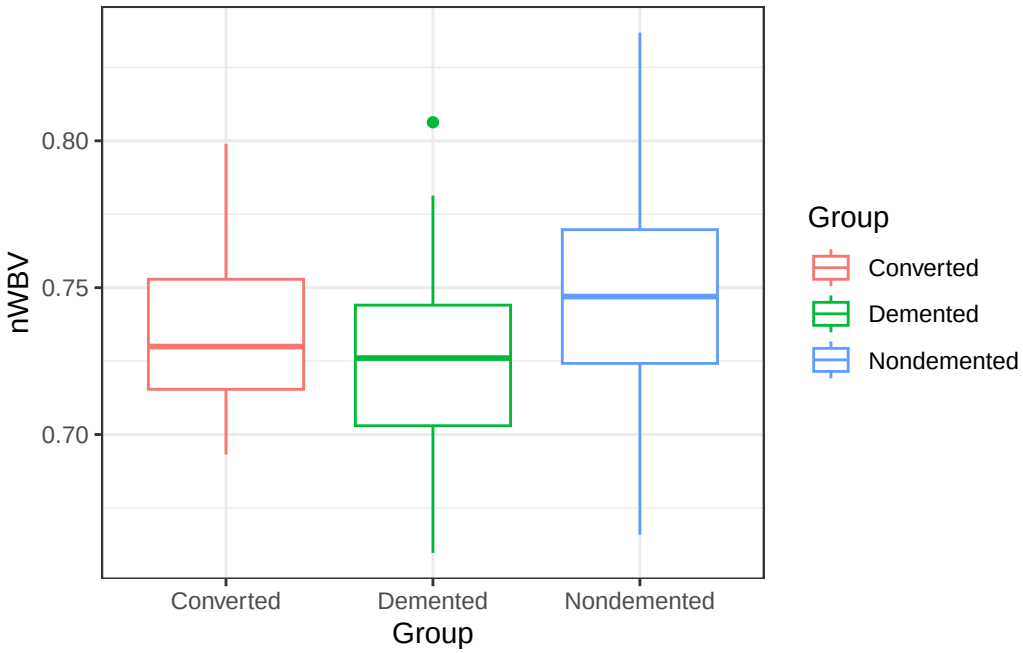


Now let's look at the grouped data.

```
# Get group means
## You can ignore this...
ybars <- aggregate(x = nWBV ~ Group, data = oasis,
                   FUN = mean)



ggplot(oasis, aes(x = nWBV, color = Group,
                  fill = Group)) +
  geom_density(alpha = 0.1) +
  geom_vline(data = ybars, aes(xintercept = nWBV,
                               color = Group))
```




```
### When comparing groups, a standard way is to use boxplots.
ggplot(oasis, aes(y = nWBV, color = Group,
                  x = Group)) +
  geom_boxplot()
```


0.80
0.75
0.70
nWBV
Converted
Demented
Nondemented
Group
Group
Converted
Demented
Nondemented

12.5.2 ANOVA in R: its `lm()` again

There are a few ways to do ANOVA in R. The approach that is most adaptable is to use the `lm()` function.

`lm(formula = y ~ x, data)`

So we create a linear model with nWBV as `y` and Group as `x`.

```
fit.lm <- lm(nWBV ~ Group, data = oasis)
```

In general, on `lm` objects we use the `summary()` function.

```
summary(fit.lm)
```

```
Call:
lm(formula = nWBV ~ Group, data = oasis)

Residuals:
      Min        1Q    Median        3Q       Max
-0.080200 -0.022459  0.000674  0.023080  0.090780

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)       0.737860   0.009421  78.317   <2e-16 ***
GroupDemented    -0.013503   0.010401  -1.298    0.196
GroupNondemented  0.008202   0.010297   0.797    0.427
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.03525 on 147 degrees of freedom
Multiple R-squared:  0.0806,    Adjusted R-squared:  0.06809
F-statistic: 6.443 on 2 and 147 DF,  p-value: 0.002078
```

This technically contains all the information we need. Specifically look at the bottom of the summary where there is information on the F-statistic, degrees of freedom and p-value.

To get this information displayed in a more traditional format for an ANOVA, use the`anova()` function on an `lm` object.

```r
anova(fit.lm)
```

```
Analysis of Variance Table

Response: nWBV
           Df   Sum Sq   Mean Sq F value   Pr(>F)
Group       2 0.016014 0.0080069  6.4431 0.002078 **
Residuals 147 0.182677 0.0012427
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

12.5.3 Alternative: `aov()`

An alternative is to use the `aov()` function in the same way:

```
fit.aov <- aov(nWBV ~ Group, data = oasis)
```

For an `aov` object, just use the `summary()` function to get results in a similar manner.

```
summary(fit.aov)
```

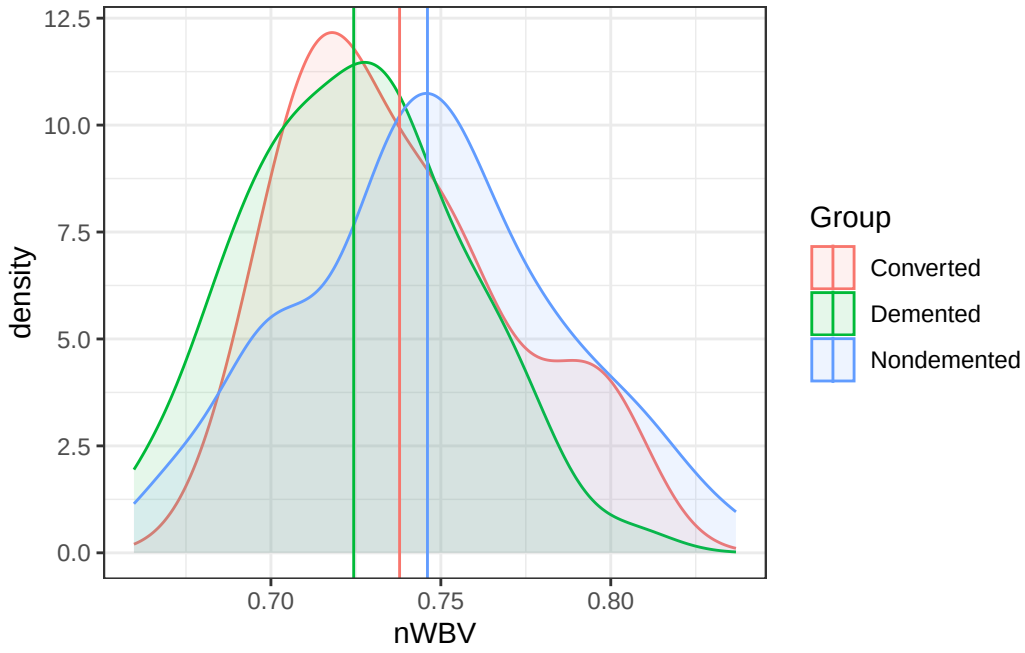
```
             Df  Sum Sq  Mean Sq F value  Pr(>F)
Group         2 0.01601 0.008007   6.443 0.00208 **
Residuals   147 0.18268 0.001243
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

In any of these cases, we have very strong discrepancy with null model (i.e., all means are equal) for a different mean nWBV between the different groups of patients.

12.5.4 Statistical Versus Practical Significance

Let's look back at the nWBVs split by group:

```
ggplot(oasis, aes(x = nWBV, color = Group,
                  fill = Group)) +
  geom_density(alpha = 0.10) +
  geom_vline(data = ybars, aes(xintercept = nWBV,
                               color = Group))
```



And here are the means of each group.

```
# This is some tidyverse magic.

oasis %>%
  group_by(Group) %>%
  summarise(mean = mean(nWBV))
```

```
# A tibble: 3 x 2
  Group        mean
  <chr>       <dbl>
1 Converted   0.738
2 Demented    0.724
```

```
3 Nondemented 0.746
```

```
# Or using the model using modelbased

library(modelbased)
fit.aov %>%
  estimate_means()
```

```
Estimated Marginal Means

Group       | Mean |       SE |       95% CI
-----------------------------------------------
Nondemented | 0.75 | 4.15e-03 | [0.74, 0.75]
Demented    | 0.72 | 4.41e-03 | [0.72, 0.73]
Converted   | 0.74 | 9.42e-03 | [0.72, 0.76]

Marginal means estimated at Group
```

Among the groups, the biggest difference in means between the demented and non-demented group is about 0.02.

Is that a big difference?

- Statistically, it is with $p = .0021$.
- On a relative scale it is about a 3% difference.
  - Is this a **practical** difference, i.e., does it matter?
  - That would be very dependent on the field, the research question, the researcher, the impacts, and so on…
  - What is an *important* difference is a specific matter that should be discussed and defined.

13 Multiple Comparisons

Here are some code chunks that setup this document.

```
# Here are the libraries I used
library(tidyverse) # standard
library(knitr)
library(readr)
library(ggpubr) # allows for stat_cor in ggplots
library(ggfortify) # Needed for autoplot to work on lm()
library(gridExtra) # allows me to organize the graphs in a grid
library(car) # need for some regression stuff like vif
library(GGally)
# library(mosaic)
```

```
# This changes the default theme of ggplot
old.theme <- theme_get()
theme_set(theme_bw())
```

Suppose we did a single test between two group means:

We would have:

$$t = \frac{\overline{y_i} - \overline{y_j}}{SE_{\overline{y}_i - \overline{y}_j}}$$

Then we get a p-value based on that t-distribution with some degrees of freedom defined by the method we were using for the comparison, e.g., equal or unequal variance two-sample t-test, ANOVA, and others…

Null Hypothesis Significance Testing (NHST) Approach

In NHST we "Reject H_0 if $p \leq \alpha$.

- α is the Type I error rate: "The probability of rejecting H_0 when H_0 is true."
  - This would be a "false positive" or a "false discovery"

P-value as a spectrum.

- We would talk about the strength of evidence for declaring $\mu_i - \mu_j$.
  - This approach would not have a Type I error per se.
  - The idea of a Type I error would fall on a spectrum as well.

13.1 Multiple testing problem

In the ANOVA setting, should we Reject H_0 (all means are equal), or declare the strength of evidence is sufficient to be confident of a difference.

- The next step is to look and see which specific groups/means differ.
- This involves pairwise comparisons.

$$t = \frac{\overline{y_i} - \overline{y_j}}{\sqrt{MSE\left(\frac{1}{n_i} + \frac{1}{n_2}\right)}}$$

- When 2 or more comparisons are done we have more than one Type I error, we have a "Family" of tests.
- This where we have the concept of **Family-Wise Error Rate (FWER)**.
  - This is defined to be the **probability of making at least one Type I error**.
  - If all the tests are "independent" then $FWER = 1-(1-\alpha_c)^k$ where α_c is the type I error rate for each individual comparison and k is the total number of comparisons in the family of tests.
- As FWER increases, the reliability of the individual comparisons decreases.
- The number of possible pairwise comparisons among t groups is $k = \frac{t(t-1)}{2}$

If we did not control the FWER error rate with $\alpha_c = 0.01$:

Table 13.1: FWER when comparing many groups via pairwise comparisons

Groups	Comparisons	FWER
2	1	0.010
3	3	0.030
4	6	0.059
5	10	0.096
6	15	0.140
7	21	0.190
8	28	0.245
9	36	0.304
10	45	0.364

There are a few of common methods for controlling FWER

Notation.

- α_F represents the desired FWER for the family of tests.
- α_C is the type I error rate for each individual test/comparison.

13.2 The Bonferroni method

Consider k hypothesis tests with $\alpha_F = 0.01$

H_{0i} vs. $H_{1i}, i = 1, \ldots, k$.

Let $p_1, \ldots, p_k$ be the p-values for these tests.

Suppose we reject H_0 when $p_i \leq \alpha_C$

$$FWER = \alpha_F \leq k \cdot \alpha_C$$

$$\alpha_C = \frac{\alpha_F}{k}$$

This is based off what is known as the Boole's theorem/inequality.

- Bonferroni was the person that connected the Boole's inequality to Hypothesis testing.
- For the more probability theory inclined people, you can see a proof at https://en.wikipedia.org/wiki/Bonferroni_correction

The Bonferroni rule for multiple comparisons is:

- Reject null hypothesis H_{0i} when $p_i < \alpha_F/k$.
- Alternatively, you could get Bonferroni corrected p-values: $p_i^* = p_i \cdot k$.
  - Reject H_0 when $p_i^* \leq \alpha_F$.
  - Use p_i^* if you are using the strength of evidence approach.

13.2.1 Example 1, OASIS data

```
oasis <- read_csv(here::here('datasets',
                             'oasis.csv'))
```

In our OASIS data, we have three groups:

- Group: whether subject is demented, nondemented, or converted.

We are trying to see if nWBV differs among the groups.

- nWBV: Normalized Whole Brain Volume; Brain volume is normalized by intercranial volume to put subjects of different sizes and gender on the same scale. To my best knowledge...

If we wanted to do all pairwise comparisons between the groups, we can do that via the `pairwise.t.test()` function.

- The form is `pairwise.t.test(x, g, p.adjust.method)`
- `x` is the response vector.
- `g` is the group vector.
- `p.adjust.method` is the method used to adjust p-values
  - There are many, but the two we will discuss are "none" and "bonferroni"
- Say we had loaded a data set called `data`, then we would idenfity the vector within the dataframe using `data$Variable`
  - `pairwise.t.test(data$responseVar, data$groupVar, "none")` would do all pairwise t-tests with no correction.
  - `pairwise.t.test(data$responseVar, data$groupVar, "bonferroni")`

There are 3 groups so there $k = \frac{3(3-1)}{2} = 3$ comparisons.

Here are the unadjusted and adjusted p-values.

```
pairwise.t.test(oasis$nWBV, oasis$Group,
                p.adjust.method = "none")
```

```
    Pairwise comparisons using t tests with pooled SD

data:  oasis$nWBV and oasis$Group
```

```
            Converted Demented
Demented    0.19624   -
Nondemented 0.42698   0.00046

P value adjustment method: none
```

```r
pairwise.t.test(oasis$nWBV, oasis$Group,
                p.adjust.method = "bonferroni")
```

```
    Pairwise comparisons using t tests with pooled SD

data:  oasis$nWBV and oasis$Group

            Converted Demented
Demented    0.5887    -
Nondemented 1.0000    0.0014

P value adjustment method: bonferroni
```

Notice the adjusted p-value is capped at 1.

13.2.2 Example, Genomics

In Genomics, 100s if not 1000s of genes are compared in terms of "activation levels" (or something like that…).

- Suppose we were comparing a 100 genes, then there would be 4950 comparisons.
  - We would multiple each unadjusted p-value by 4950 which means and individual comparison would have to have an unadjusted p-value less than 0.00010 to be rejected with $\alpha_F = 0.05$ using the Bonferroni method.
  - We would probably miss many results where there is a difference in mean activation levels.
- The Bonferroni method is overly conservative, i.e., is fails to reject the null hypothesis too often (it has low power).
- In general, the Bonferroni method **should not** be applied unless you absolutely have to (which I cannot think of a situation where you would except for your "boss" telling you to).
- However, you are still expected to know it, so here it is.

13.3 Tukey's HSD (Tukey's Honestly Significant Difference)

$$MSE = \frac{SSE}{N-k}$$

$$Q = \frac{|\max_i(\overline{y}_i) - \min_i(\overline{y}_j)|}{\sqrt{\frac{2 \cdot MSE}{n}}}$$

Under the null hypothesis of ANOVA (all means equal) then Q is a random variable of the "Studentized Range Distribution", $q(t, N-t)$.

The Tukey method does not *have* to be used in ANOVA, but that is the main setting where it applies.

13.3.1 Tukey in R

Now a pain with R is all of these different types of analysis methods, e.g., `lm`, `aov`, etc., may have different subsidiary functions to into further analysis.

By default, `TukeyHDS()` is a very convenient function for doing Tukey comparisons. However, it only works on `aov` objects.

```
fit.lm <- lm(nWBV ~ Group, oasis)
fit.aov <- aov(nWBV ~ Group, oasis)


# TukeyHSD(fit.lm) # THIS GIVES AN ERROR

TukeyHSD(fit.aov)
```

```
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = nWBV ~ Group, data = oasis)

$Group
                              diff          lwr        upr     p adj
Demented-Converted    -0.013502877 -0.038129366 0.01112361 0.3984496
Nondemented-Converted  0.008202274 -0.016177415 0.03258196 0.7057132
Nondemented-Demented   0.021705151  0.007366034 0.03604427 0.0013286
```

I personally think that's stupid. `lm` and `aov` are two sides of the same analysis coin. All relevant calculations are the same, just presented in different ways.

The `modelbased` or `emmeans` packages can help resolve these problems.

Anyway…

```
library(emmeans)
emmeans(fit.lm,
        pairwise ~ Group,
        adjust = "tukey")
```

```
$emmeans
 Group       emmean      SE  df lower.CL upper.CL
 Converted    0.738 0.00942 147    0.719    0.756
 Demented     0.724 0.00441 147    0.716    0.733
 Nondemented  0.746 0.00415 147    0.738    0.754

Confidence level used: 0.95

$contrasts
 contrast                estimate      SE  df t.ratio p.value
 Converted - Demented      0.0135 0.01040 147   1.298  0.3984
 Converted - Nondemented  -0.0082 0.01030 147  -0.797  0.7057
 Demented - Nondemented   -0.0217 0.00606 147  -3.584  0.0013

P value adjustment: tukey method for comparing a family of 3 estimates
```

13.4 FDR and the Benjamani-Hochberg procedure

There are four scenarios when it comes to the Reject/Fail to reject testing procedure.

Test Result	Alternmative is true	Null hypothesis true	Total
Tests significant	TP	FP	P
Test is declared non-significant	FN	TN	N
Total	$m - m_0$	m_0	m

False Discovery Rate: The proportion of times that you reject the null hypothesis when it is true, i.e., the proportion of Type I Errors.

$$FDR = \frac{FP}{FP + TP} = \frac{FP}{P}$$

13.4.1 Controlling the FDR: Benjamani-Hochberg Procedure

1. Specify an acceptable maximum false discovery rate q (or α)
2. Order the p-values from least to greatest: $p_{(1)}, p_{(2)}, \dots, p_{(m)}$.
3. Find k, which is the largest value of i such that $p_{(i)} \leq \frac{i}{m}q$, i.e, $k = \mathsf{max}\{i : p_{(i)} \leq \frac{i}{m}q\}$
4. Reject all null hypotheses corresponding to the p-values $p_{(1)}, \dots, p_{(k)}$. That is, reject the k-th smallest p-values.

```
pairwise.t.test(oasis$nWBV, oasis$Group, p.adjust.method = "BH")
```

```
    Pairwise comparisons using t tests with pooled SD

data:  oasis$nWBV and oasis$Group

            Converted Demented
Demented    0.2944    -
Nondemented 0.4270    0.0014

P value adjustment method: BH
```

13.4.2 Other Procedures for Controlling FDR

There are a couple methods for controlling FDR in `pairwise.t.test()`:

- Benjamani-Hochberg: `p.adjust.method = "BH"` (you may also put `"fdr"` instead of `"BH"`)
- Benjamani-Yekutieli: `p.adjust.method = "BY"`

14 ANOVA Assumptions

Here are some code chunks that setup this chapter.

```
# Here are the libraries I used
library(tidyverse) # standard
library(knitr)
library(readr)
library(ggpubr) # allows for stat_cor in ggplots
library(ggfortify) # Needed for autoplot to work on lm()
library(gridExtra) # allows me to organize the graphs in a grid
library(car) # need for some regression stuff like vif
library(GGally)
library(mosaic) # For nicer TukeyHSD function that works with lm()
```

```
# This changes the default theme of ggplot
old.theme <- theme_get()
theme_set(theme_bw())
```

14.1 Notation Reminder

- y_{ij} is the j^{th} observation in the i^{th} treatment group.
  - $i = 1, \ldots, t$, where t is the total number of treatment groups.
  - $j = 1, \ldots, n_i$ where n_i is the number of observations in treatment group i.
- $\overline{y}_{i.} = \sum_{j=1}^{n_i} \frac{y_i j}{n_i}$ is the mean of treatment group i.
- $\bar{y}_{..} = \sum_{i=1}^{t} \sum_{j=1}^{n_i} \frac{y_{ij}}{N}$ is the over all mean of all observations.
  - N is the total number of observations.

14.2 Assumptions

In ANOVA, we have the same assumptions. They have to do with the residuals!

Before the residuals in linear regression were:

$$e_i = y_i - \widehat{y}_i$$

Well now, an observations is y_{ij} and the any observation would best be predicted by its group mean $\overline{y}_i$

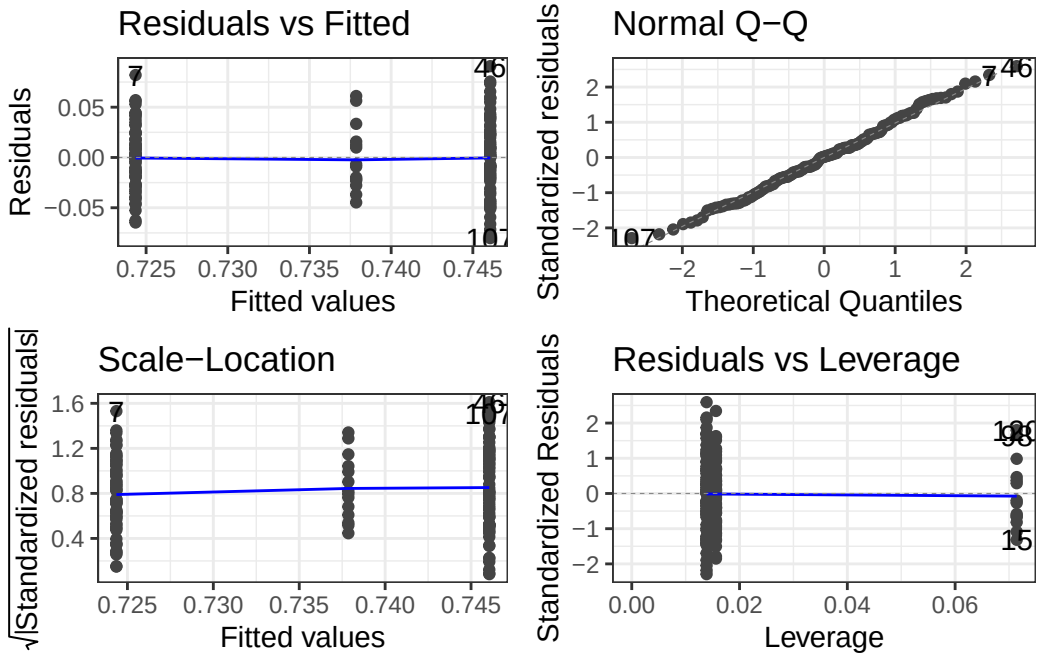
$$e_i = y_{ij} - \overline{y}_{i.}$$

- We assume the residuals are still normally distributed.
- The variability is constant between groups.
  - Denote the standard deviation of population group i with σ_i.
  - We assume $\sigma_1 = \sigma_2 = \cdots = \sigma_t$.
- We assume they are independent. This is an issue in the situation where measurements are recorded over time.
- In linear regression, we additionally had to worry about model bias.
  - This is not an issue in ANOVA.
  - This is because we are estimating group means using sample means. Sample means are unbiased estimators by their very nature.

14.3 Checking them is about the same! `autoplot()`

```
oasis <- read_csv(here::here("datasets",
                             'oasis.csv'))
```

```
fit.lm <- aov(nWBV ~ Group, oasis)

autoplot(fit.lm)
```




```
oasis %>%
  group_by(Group) %>%
  summarise(SD = sd(nWBV),
            Mean = mean(nWBV),
            n = length(nWBV))
```

```
# A tibble: 3 x 4
  Group           SD  Mean     n
  <chr>        <dbl> <dbl> <int>
1 Converted   0.0330 0.738    14
2 Demented    0.0315 0.724    64
3 Nondemented 0.0387 0.746    72
```

The variability looks a bit smaller in the middle group, i.e., the Converted group. HOWEVER, that is only because there are so few observations in that group.

The normality looks pretty good.

14.3.1 Testing for Constant Variability/Homoskedasticity: Levene's Test and Brown-Forsythe Test

To test for for constant variability in ANOVA, we use ANOVA.

The hypotheses are

$H_0 : \sigma_1 = \sigma_2 = \cdots = \sigma_t$ H_1 : At least one difference exists.

Instead of using the observations y_{ij}, we use

$$z_{ij} = |y_{ij} - \overline{y}_{i.}|$$

or

$$z_{ij} = |y_{ij} - \tilde{y}_{i.}|$$

where $\tilde{y}_{i.}$ is the median of a group.

- We call it **Levene's Test** when using the mean $\overline{y}_{i.}$ to compute z_{ij}.
- We call it the **Brown-Forsythe Test** when using the median $\tilde{y}_i$ to compute z_{ij}.

Then the ANOVA process is used to to see if the mean of the z values differ between the groups.

- $SST^* = \sum_{i=1}^{n} n_i \left(\overline{z}_{i.} - \overline{z}_{..}\right)^2$
- $SSE^* = \sum_{i=1}^{n} \left(z_{ij} - \overline{z}_{i.}\right)^2$

$\overline{z}_{i.}$ and $\overline{z}_{..}$ are the group means and overall mean of the z_{ij}'s.

And then we have the mean squares. Just as before.

- $MST^* = SST/(t-1)$
- $MSE^* = SSE/(N-t)$

And our test statistic is

$$F_t^* = MST^*/MSE^*$$

- Under the null hypothesis this test statistic followsan $F(t-1, N-t)$ distribution which is used to compute its p-value.

FYI: In many texts that I have seen, it is sometimes referred to as W instead of F_t^*.

14.3.2 Levene/Brown-Forsythe in R

You need the `car` library and the function for performing either the Levene or Brown-Forsythe in R is `leveneTest()`

You only need one argument, your `lm()` or `aov()` depending on how you do the ANOVA:

- `leveneTest(model)`
- Despite its name, it is doing the Brown-Forsythe version of the test by default.
- To do the Levene Test, you have to specify a `center` argument.
  - `center = median` is Brown-Forsythe by default.
  - `center = mean` is Levene.
  - So to perform Levene's Test: `leveneTest(model, center = mean)`

14.3.3 Oasis Example

```
fit.aov <- aov(formula = nWBV ~Group,
               data=oasis)



car::leveneTest(fit.aov)
```

```
Levene's Test for Homogeneity of Variance (center = median)
       Df F value Pr(>F)
group   2  1.0268 0.3607
      147
```

You may see some sort of warning message like:

```
Warning message:
In leveneTest.default(y = y, group = group, ...) : group coerced to factor.
```

Technically, the group variable should be a "factor" type of variable in `R`. This is just telling your grouping variable isn't a "factor" type of variable and the function is assuming that it should be a "factor".

`leveneTest` works on both the `aov()` and `lm()` model types.

14.4 What if the assumptions are violated?

- Should normality be a major issue, then either transformations should be attempted or you should look into something called the Kruskal-Wallis test. And probably consult a statistician.
- If the variability is not constant across groups, then there is Welch version of the ANOVA test.

To perform the Welch ANOVA, you would use the `oneway.test()`.

You input just like with `lm()` or `aov()`.

```
oneway.test(nWBV ~ Group, oasis)
```

```
    One-way analysis of means (not assuming equal variances)

data:  nWBV and Group
F = 6.4889, num df = 2.000, denom df = 37.508, p-value = 0.003801
```

Compare that to the standard ANOVA:

```
summary(fit.aov)
```

```
             Df  Sum Sq  Mean Sq F value  Pr(>F)
Group         2 0.01601 0.008007   6.443 0.00208 **
Residuals   147 0.18268 0.001243
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

There is not much of a difference since the Brown-Forsythe test did not indicate there was reason to conclude the variability isn't constant.

14.4.1 Games-Howell Procedure

The Games-Howell procedure is used to do pairwise comparisons when normality is not an issue but homogeneity is a concern.

- It is available via the `rstatix` package.
- Use the `games_howell_text()` function.

The format is `games_howell_test(data, formula, conf.level = 0.95, detailed = FALSE)`.

- `data`: a data.frame containing the variables in the formula.
- `formula`: a formula of the form x ~ group where x is a numeric variable giving the data values and group is a factor with one or multiple levels giving the corresponding groups. For example, `formula = TP53 ~ cancer_group`.
- `conf.level`: confidence level of the interval.
- `detailed`: logical value. Default is FALSE. If TRUE, a detailed result is shown.

This package intends to change the paradigm of functions such that they are formatted so that the `data` argument is first. This is meant for compatibility with newer data-sciency uses of `R` that involve things called "pipes"

14.5 Some Extra Remarks

From this ResearchGate question.

Bruce Weaver writes what I consider to be some rather nice advice.

The numbered parts are verbatim from that link:

1. Levene's test is a test of homogeneity of variance, not normality.
2. Testing for normality as a precursor to a t-test or ANOVA is not very helpful, IMO. Normality (within groups) is most important when sample sizes are small, but that is when tests of normality have very little power to detect non-normality. As the sample sizes increase, the sampling distribution of the mean converges on the normal distribution, and normality of the raw scores (within groups) becomes less and less important. But at the same time, the test of normality has more and more power, and so will detect unimportant departures from normality.
3. Rather than testing for normality, I would ask if it is fair and reasonable to use means and SDs for description. If it is, then ANOVA should be fairly valid.
4. Many authors likewise do not recommend testing for homogeneity of variance prior to doing a t-test or ANOVA. Box said that doing a preliminary test of variances was like putting out to sea in a row boat to see if conditions are calm enough for an ocean liner. I.e., he was saying that ANOVA is very robust to heterogeneity of variance. That is especially so when the sample sizes are equal (or nearly so). But as the sample sizes become more discrepant, heterogeneity of variance becomes more problematic. When sample sizes are reasonably similar, some authors suggest that ANOVA is robust to heterogeneity of variance so long as the ratio of largest variance to smallest variance is no more than 5:1.

However, let me add a note:

- It may not necessarily be useful to *test* for homogeneity of variability (and other assumptions), but it is good to inspect plots of the assumptions. If there is a huge discrepancy, then you're going to want to address them.

14.5.1 One Final Note: Sample Sizes

Hopefully you end up taking a class in "Experimental Design".

- There will be an emphasis on experiments with "balanced" data, i.e., each group has equal sample size.
- Often this will be impossible.
- Especially in experiments involving people.

- I have tried to provide tools that are robust alternatives.
  - A caveat is that if you have extremely unbalanced samples, e.g., 5 in one group and 50 in another group.
  - You should be very careful and probably need to learn new methods or consult a statistician.

15 Balanced (Uniform Sample Size) Two-Way ANOVA

Here are some code chunks that setup this chapter

```
# Here are the libraries I used
library(tidyverse) # standard
library(knitr)
library(readr)
library(ggpubr) # allows for stat_cor in ggplots
library(ggfortify) # Needed for autoplot to work on lm()
library(gridExtra) # allows me to organize the graphs in a grid
library(car) # need for some regression stuff like vif
library(GGally)
library(emmeans)
```

```
# This changes the default theme of ggplot
old.theme <- theme_get()
theme_set(theme_bw())
```

The idea of analysis of variance mainly stems from concepts of "experimental design" which is a separate course that should be taken should you be in a field that heavily focuses on, well... quantitative experiments that will be analyzed.

There is so much there that we can't even touch on it in a meaningful way with the scope of this course.

Anyway, in an experiment we manipulate external variables and see their affect on the response variable.

- y is our response variable
- We can have various x variables that we are manipulating.
  - In ANOVA context these are called **factors**.
  - We can have more than one factor. (Technically as many as we want, but probably stop at 3!)
  - Each factor has a set of levels, i.e. the specific values that are set

  - A **treatment** is the specific combination of the levels of all the factors.
  - Each individual treatment could be referred to as a **cell**.

- Group A: A_1_,A_2_,A_3_
- Group B: B_1_, B_2_
- Group Combinations: A_1_B_1_,A_2_B_1_,A_3_B_1_,A_1_B_2_,A_2_B_2_,A_3_B_2_

15.1 Pseudo-Example

We are doing an ANOVA type experiment.

- We have are subjects and we want to test the effectiveness of a drug and exercise regime on blood pressure.
- The drug is factor A and there will be 3 dose **levels**: 0mg (control), 5mg, and 10mg
- The exercise regime is factor B with 3 **levels**: none (control), light, heavy.
- This would be referred to as a 3 by 3 (3×3) ANOVA.
  - In general we say $A \times B$ where we put the number of levels for each factor.
  - In this case there would be 9 total unique combinations of the two factors which means there would be 9 **treatments**

We can view it as a grid kind of pattern for the design of the experiement.

Regime\Dose	0mg	5mg	10mg
none	0 & none (control)	5 & none	10 & none
light	0 & light	5 & light	10 & light
heavy	0 & heavy	5 & heavy	10 & heavy

And there would be a random assignment of each treatment to a set of subjects.

15.1.1 Sample Sizes in Two-Way ANOVA

- A basic analysis is possible if there is only one subject per treatment (cell).
  - This is not an ideal scenario.
- More than 1 subject per treatment is better.
- An equal number of subjects for each treatment is "best", which is called a **balanced design**
  - This may not feasible many times when involving living individuals (humans or otherwise.

  - Subject smay drop out of studies for various reasons.
  - Levels of factors may have different difficulties for obtaining/producing.
  - Variability of subject availability depending on treatments.
  - Etc.

- If there are not an equal number of individuals within each group, it is referred to as an **unbalanced design**.
- This *usually* not a problem with a large enough sample size and when there isn't heteroscedasticity (check your assumptions!)

We will cover Balanced Design.

Unbalanced Design requires much more care and you should consult someone with experience.

Should time allow, we will cover it. But it really is just a mess to wrap your head around.

15.2 Notation and jargon

There's a lot here. The notation logic is fairly procedural (to a degree that it becomes confusing).

Factors and Treatments

We have factor A with levels $i = 1, 2, \dots, a$ and factor B with levels $j = 1, 2, \dots, b$.

- A_i represents level i of factor A.
- B_j represents level j of factor B.
- A_iB_j represents the treatment combination of level i and level j of factors A and B respectively.
- This is sometimes referred to as "treatment ij".

Observations and Sample Means

An individual observation is denoted by y_{ijk}. The subscripts indicate we are looking at observation k from treatment A_iB_j

- There are n observations in each individual treatment treatment, $k = 1, 2, \dots, n$.
- $\overline{y}_{ij.}$ is the mean of the observations in treatment A_iB_j. ($\overline{y}_{ij.} = \frac{1}{n}\sum_{k=1}^{n} y_{ijk}$)
- $\overline{y}_{i..}$ is the mean of the observations in treatment A_i across all levels of factor B. ($\overline{y}_{i..} = \frac{1}{bn}\sum_{j=1}^{b}\sum_{k=1}^{n} y_{ijk}$)
- $\overline{y}_{.j.}$ is the mean of the observations in treatment B_j across all levels of factor A. ($\overline{y}_{.j.} = \frac{1}{an}\sum_{i=1}^{a}\sum_{k=1}^{n} y_{ijk}$)
- $\overline{y}_{...}$ is the mean of all observations. ($\overline{y}_{...} = \frac{1}{abn}\sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{k=1}^{n} y_{ijk}$)

15.3 Two way ANOVA Model

There is a **means model**:

$$y = \mu_{ij} + \epsilon$$

- μ_{ij} is the mean of treatment A_iB_j.
- Epsilon is the error term and it is assumed $\epsilon \sim N(0, \sigma)$.
  - The constant variability assumption is here.

Which in my opinion is not congruent with what we are trying to investigate in Two-Way ANOVA.

- We are trying to understand the effect that both factors have on the the response variable.
- There may be an **interaction** between the factors.
  - This is when the effect the levels of factor A is not consistent across all levels of factor B, and vice versa.
  - For example, the mean of the response variable may increase when going from treatment A_1B_1 to A_2B_1, but the response variable mean decreases when we look at A_1B_2 to A_2B_2.

With this all in mind, the **effects model** in two-way ANOVA is:

$$y = \mu + \alpha_i + \beta_j + \gamma_{ij} + \epsilon$$

- μ would be the mean of the response variable without the effects of the factors.
  - This can usually be thought of as the mean of a control group.
- α_i can be thought of as the effect that level i of factor A has on the mean of y.
- Likewise, β_j is the effect that level j of factor B has on the mean of y.
- γ_{ij} is the interaction effect, which is the additional effect that the specific combination A_i and B_j has on the mean of y.

15.3.1 Estimating model parameters (Means model)

If we are considering a means model there are a few things we can consider:

- The overall mean of the data $\mu_{..}$ which is estimated by $\overline{y}_{...}$
- The overall mean of A_i (when averaged over factor B) $\mu_{i.}$ which is estimated by $\overline{y}_{i..}$
- The overall mean of B_j (when averaged over factor B) $\mu_{.j}$ which is estimated by $\overline{y}_{.j.}$
- The treatment mean of A_iB_j μ_{ij} which is estimated by $\overline{y}_{ij.}$

15.3.2 Estimating model parameters (Effects model)

It might be more interesting to estimate how big the effect is of a given treatment or how large an interaction is.

- I would argue that the only meaningfully done IF there is a control group in both factors.

Assume that A_1 and B_1 are the control levels.

- Then α_1 and β_1 should be (at least conceptually) 0.
- There would be no interaction terms for any treatment A_1B_j or A_iB_j
- $\hat{\alpha}_i = \overline{y}_{i..} - \overline{y}_{1..}$
- $\hat{\beta}_i = \overline{y}_{.j.} - \overline{y}_{.1.}$
- $\hat{\gamma}_{ij} = \overline{y}_{ij.} - \overline{y}_{i1.} - \overline{y}_{1j.} + \overline{y}_{ij.}$

Estimating the interactions involves things we call *contrasts* and that is another can of worms.

15.4 Hypothesis Tests

There are three sets of hypotheses that can potentially be tested in ANOVA.

15.4.1 Main Effects Tests

We can test whether factor A has an effect.

$H_0 : \alpha_i = 0$ for all i, i.e., factor A has no effect on the mean of y.

- Note that this would be the hypothesis if there were a control group.
- Otherwise it would be more correct to say that all levels of factor A have an equivalent effect ($\alpha_1 = \alpha_2 = \cdots = \alpha_a$)

$H_1 : \alpha_i \neq 0$ for at least one i. At least one level of factor A has an effect on the mean of y.

- Or without a control, at least one α_i differs from the rest.

Likewise we can perform a separate test for whether factor B has an effect:

$H_0 : \beta_j = 0$ for all j, i.e., factor B has no effect on the mean of y.

- Note that this would be the hypothesis if there were a control group.
- Otherwise it would be more correct to say that all levels of factor B have an equivalent effect ($\beta_1 = \beta_2 = \cdots = \beta_b$)

$H_1 : \beta_j \neq 0$ for at least one j. At least one level of factor B has an effect on the mean of y.

- Or without a control, at least one β_i differs from the rest.

15.4.2 Interaction Test

BEFORE Before you can rely on the tests for the main effects, you must first consider testing for whether there is any meaningful interaction.

H_0: $\gamma_{ij} = 0$ for all A_iB_j treatment combinations.

- The idea of changing the hypothesis for whether there is a control or not is a bit more technical.
- You *could* say the effects are all equivalent but then there really isn't an interaction.
- There are multiple valid hypotheses IMO. So stick with this one to avoid getting too technical.

$H_1 : \gamma_{ij} \neq 0$ for at least on A_iB_j treatment combination.

The reason that this test must be the first one to be considered is that if there is an interaction effect, it becomes mathematically impossible to effectively conclude the strength of an effect of an individual factor overall.

- Interaction implies the effect of one factor changes depending on another factor so statements about individual factors are a moot point.

15.4.3 Sums of Squares, Mean Squares, and Test Statistics

We are going to ignore all the formula based stuff at this point, just consider the concepts.

- We measure how meaningful a factor or interaction is via sums of squares, just as before.
- SSA measures how meaningful is factor A within the data.
  - The degrees of freedom is $a - 1$.
- SSB measures how meaningful is factor B within the data.
  - The degrees of freedom is $b - 1$
- $SSAB$ measures how meaningful is the interaction within the data.
  - The degrees of freedom is $(a - 1)(b - 1)$
- SSE is the measure of how much variability is left unexplained by the factors and interaction.
  - The degrees of freedom is $ab(n - 1)$

"Meaningful" translates to "the amount of variability in data the data that is explained by including the variable in the model".

Just as before, to get test statistics we need to compute mean squares.

- $MSA = SSA/(a-1)$
- $MSB = SSB/(b-1)$
- $MSAB = SSAB/(a-1)(b-1)$
- $MSE = SSE/(ab(n-1))$

And then we get F test statistics.

- $F_A = MSA/MSE$ tests whether factor A has an effect, $H_0 : \alpha_i = 0$ for all i.
- $F_B = MSB/MSE$ tests whether factor B has an effect, $H_0 : \beta_j = 0$ for all j.
- $F_{AB} = MSAB/MSE$ tests whether there is an interaction effect, H_0: $\gamma_{ij} = 0$ for all i and j.

15.4.4 And so there is a Two-Way ANOVA table (surprise)

Source	SS	df	MS	F	p
Factor A	SSA	$a-1$	$MSA = SSA/(a-1)$	$F_A = MSA/MSE$	p_A
Factor B	SSB	$b-1$	$MSB = SSB/(b-1)$	$F_B = MSB/MSE$	p_B
Interaction: A*B	$SSAB$	$(a-1)(b-1)$	$MSAB = SSAB/((a-1)(b-1))$	$F_{AB} = MSAB/MSE$	p_{AB}
Error	SSE	$ab(n-1)$	$MSE = SSE/(ab(n-1))$		

- Reject H_0 if $p < \alpha$
- $P_A \rightarrow H_0 : \alpha_i = 0$
- $P_A \rightarrow H_0 : \beta_j = 0$
- $P_{AB} \rightarrow H_0 : \gamma_i = 0$

15.5 Study: Compulsive Checking and Mood

This data is taken from the textbook resources of: Discovering Statistics Using R by Andy Field, Jeremy Miles, Zoë Field

which itself is using data from a journal article (one textbook author is a paper co-author):

The perseveration of checking thoughts and mood–as–input hypothesis by Davey et al. (2003), https://doi.org/10.1016/S0005-7916(03)00035-1.

The study investigated the relation between mood and when people will stop performing a task under different stopping rules. They were trying to explore the connection with Obsessive Compulsive Disorder.

- There were 60 participants (it was one of those college student studies).
- The mood of a participant was "induced" via music:
  - 20 were assigned to have a "negative" mood induction.
  - 20 were assigned to have a "positive" mood induction.
  - 20 were assigned to have a "neutral" mood induction.
- Participants were then asked to "to write down all the things they would wish to check for safety or security reasons in their home before they left for a 3-week holiday".
- Within each mood group:
  - 10 participants were told to continue the task only if they felt like continuing.
  - 10 participants were told to continue the task until they've listed as many as they can.

Data are available in `compulsion.csv`.

- `items`: how many items a participant listed.
- `mood`: which mood induction group the participant was in.
  - `Negative`
  - `Positive`
  - `Neutral`
- `stopRule`: what was the stopping rule
  - `Many` : "As many as you can"
  - `Feel` : "Feel like continuing"

```
ocd <- read_csv(here::here("datasets",'ocd.csv'))
```

Here is what a sample of the data look like

```
# A tibble: 10 x 3
   items mood     stopRule
   <dbl> <chr>    <chr>
 1     7 Negative Many
 2     5 Neutral  Feel
 3    10 Neutral  Many
```

```
 4     8 Positive Feel
 5    15 Negative Feel
 6     5 Negative Many
 7    13 Negative Many
 8    14 Neutral  Feel
 9    31 Positive Feel
10    14 Positive Feel
```

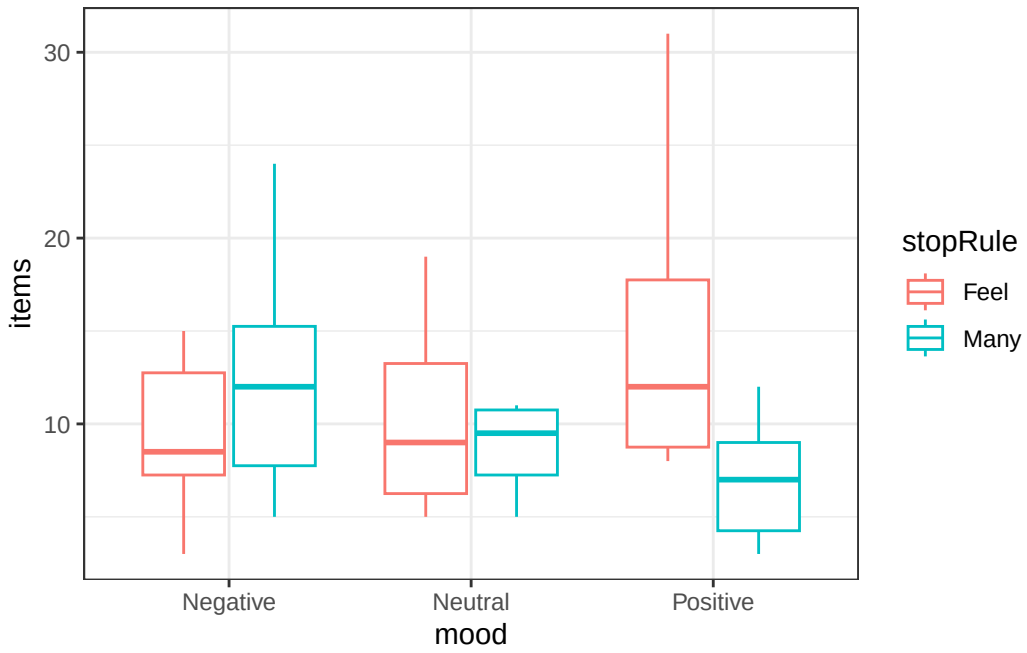
For videos related to the analysis of this data in R, I'll go over some issues with the analyses performed. We are basically following along with what was done in the paper.

15.5.1 Examining the data

Always take a look at your data first.

We'll look at boxplots, you can group by mood and color by stopping rule.

```
ggplot(ocd, aes(y = items, x = mood, color = stopRule)) +
  geom_boxplot()
```



15.5.2 Performing two-way ANOVA

ANOVA is similar to before and we can add variables like in regression. There is a trick to adding in an interaction.

You specify the interaction of two variables by "multiplying" them with the `:` symbol in the formula, e.g., `y ~ x + z + x:z`

For two-way ANOVA, I highly recommend always using the `Anova()` function from the `car` package, not the `Anova()` function. The reason will be clarified

```
ocd.fit <- lm(items ~ mood + stopRule + mood:stopRule,
              data = ocd)

car::Anova(ocd.fit)
```

```
Anova Table (Type II tests)

Response: items
               Sum Sq Df F value   Pr(>F)
mood            34.13  2  0.6834 0.509222
stopRule        52.27  1  2.0928 0.153771
mood:stopRule  316.93  2  6.3452 0.003349 **
Residuals     1348.60 54
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Equivalently, you could "multiply" via the `*` symbol without listing the individual variables.

- The `*` tells R to include all individual AND interaction terms when used in a formula, e.g., `y ~ x*z`

So equivalently we could do:

```
ocd.fit2 <- lm(items ~ mood*stopRule, data = ocd)
Anova(ocd.fit2)
```

```
Anova Table (Type II tests)

Response: items
               Sum Sq Df F value   Pr(>F)
mood            34.13  2  0.6834 0.509222
stopRule        52.27  1  2.0928 0.153771
```

```
mood:stopRule  316.93  2  6.3452 0.003349 **
Residuals     1348.60 54
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

And we get equivalent results.

15.6 Post-hoc Comparisons: Estimated Marginal Means

Depending how we want to view the data, the means we are interested in may change.

- If we want to (and can) investigate the main effects, we look at what are referred to as the **marginal means**
  - $\overline{y}_{i..}$ is the marginal mean of level i of factor A. (Averaged across all levels of B)
  - $\overline{y}_{.j.}$ is the marginal mean of level j of factor B. (Averaged across all levels of A)
- $\overline{y}_{ij.}$'s are known as the cell or treatment means. The mean of treatment A_iB_j

15.6.1 Using the `emmeans()` function to get marginal or cell means

The `emmeans()` function within the eponymous `emmeans` package.

We will stick with the simplest way.

- The format is `emmeans(model, specs, level = 0.95)`
- `model` is the `lm` or other type of model you create.
- `specs` specifies what means you want and how you want to adjust them.
  - The input is a formula style but without the response variable listed, i.e., `~ x` and **not** `y ~ x` (unless you know some more "fun" stuff).
  - `~ A` will compute the marginal means for each level of factor A. This should only be used if an interaction term is not "significant".
  - `~ A | B` will calculate the cell means for the levels of factor A conditioned upon what the level of factor B is.
  - `~ A | B` is a way of organizing the interaction means in a Two-Way model
  - You can switch which variable. `B | A`.
  - When you specify at condition variable using `|` you are getting only the a portion of the interaction means.
  - `~ A*B` will compute the cell/interaction means
- `level` specifies the confidence level of resulting confidence interval
- There is an `adjust` argument, it is `tukey` by default and lets just leave it like that.

- `adjust` with change automatically depending on the situation.

```
emmeans(ocd.fit, ~ mood, level = 0.99)
```

```
 mood     emmean   SE df lower.CL upper.CL
 Negative   10.9 1.12 54     7.92     13.9
 Neutral     9.3 1.12 54     6.32     12.3
 Positive   10.9 1.12 54     7.92     13.9

Results are averaged over the levels of: stopRule
Confidence level used: 0.99
```

```
emmeans(ocd.fit, ~ mood | stopRule, level = 0.99)
```

```
stopRule = Feel:
 mood     emmean   SE df lower.CL upper.CL
 Negative    9.2 1.58 54     4.98     13.4
 Neutral     9.9 1.58 54     5.68     14.1
 Positive   14.8 1.58 54    10.58     19.0

stopRule = Many:
 mood     emmean   SE df lower.CL upper.CL
 Negative   12.6 1.58 54     8.38     16.8
 Neutral     8.7 1.58 54     4.48     12.9
 Positive    7.0 1.58 54     2.78     11.2

Confidence level used: 0.99
```

```
emmeans(ocd.fit, ~ mood:stopRule, level = 0.99)
```

```
 mood     stopRule emmean   SE df lower.CL upper.CL
 Negative Feel        9.2 1.58 54     4.98     13.4
 Neutral  Feel        9.9 1.58 54     5.68     14.1
 Positive Feel       14.8 1.58 54    10.58     19.0
 Negative Many       12.6 1.58 54     8.38     16.8
 Neutral  Many        8.7 1.58 54     4.48     12.9
 Positive Many        7.0 1.58 54     2.78     11.2

Confidence level used: 0.99
```

Note that it warns you that an interaction is present and therefore you should not look at single factor means. (How convenient.)

15.6.2 Getting pairwise comparisons

To get pairwise comparisons, you save your `emmeans()` as a variable and pass it to the `pairs()` function.

The format is 'pairs(emmeansThing, adjust = "tukey")

- `adjust` will adjust resulting confidence intervals or hypothesis tests based on any of the methods discussed.
  - "tukey" is an option, use this option when doing pairwise comparisons unless you have unbalanced data
  - "sidak" is another option.
  - there are many more

```
moodMeans <- emmeans(ocd.fit, ~ mood, level = 0.99)
stopRuleBymoodMeans <- emmeans(ocd.fit, ~ stopRule | mood, level = 0.99)
interactionMeans <- emmeans(ocd.fit, ~ mood:stopRule, level = 0.99)

pairs(moodMeans, adjust = "tukey")
```

```
 contrast            estimate   SE df t.ratio p.value
 Negative - Neutral       1.6 1.58 54   1.012  0.5723
 Negative - Positive      0.0 1.58 54   0.000  1.0000
 Neutral - Positive      -1.6 1.58 54  -1.012  0.5723

Results are averaged over the levels of: stopRule
P value adjustment: tukey method for comparing a family of 3 estimates
```

```
pairs(stopRuleBymoodMeans, adjust = "tukey")
```

```
mood = Negative:
 contrast    estimate   SE df t.ratio p.value
 Feel - Many     -3.4 2.23 54  -1.521  0.1340

mood = Neutral:
 contrast    estimate   SE df t.ratio p.value
 Feel - Many      1.2 2.23 54   0.537  0.5935

mood = Positive:
 contrast    estimate   SE df t.ratio p.value
 Feel - Many      7.8 2.23 54   3.490  0.0010
```

```
pairs(interactionMeans, adjsut = "tukey")
```

```
 contrast                       estimate   SE df t.ratio p.value
 Negative Feel - Neutral Feel       -0.7 2.23 54  -0.313  0.9996
 Negative Feel - Positive Feel      -5.6 2.23 54  -2.506  0.1406
 Negative Feel - Negative Many      -3.4 2.23 54  -1.521  0.6523
 Negative Feel - Neutral Many        0.5 2.23 54   0.224  0.9999
 Negative Feel - Positive Many       2.2 2.23 54   0.984  0.9210
 Neutral Feel - Positive Feel       -4.9 2.23 54  -2.192  0.2582
 Neutral Feel - Negative Many       -2.7 2.23 54  -1.208  0.8310
 Neutral Feel - Neutral Many         1.2 2.23 54   0.537  0.9944
 Neutral Feel - Positive Many        2.9 2.23 54   1.298  0.7850
 Positive Feel - Negative Many       2.2 2.23 54   0.984  0.9210
 Positive Feel - Neutral Many        6.1 2.23 54   2.729  0.0859
 Positive Feel - Positive Many       7.8 2.23 54   3.490  0.0118
 Negative Many - Neutral Many        3.9 2.23 54   1.745  0.5090
 Negative Many - Positive Many       5.6 2.23 54   2.506  0.1406
 Neutral Many - Positive Many        1.7 2.23 54   0.761  0.9729

P value adjustment: tukey method for comparing a family of 6 estimates
```

Note the p-values are adjusted depending on how many groups are being compared:

```
mood = Positive:
 contrast                                    estimate   SE df t.ratio p.value
 As many as you can - Feel like continuing     -7.8 2.23 54 -3.490  0.0010
```

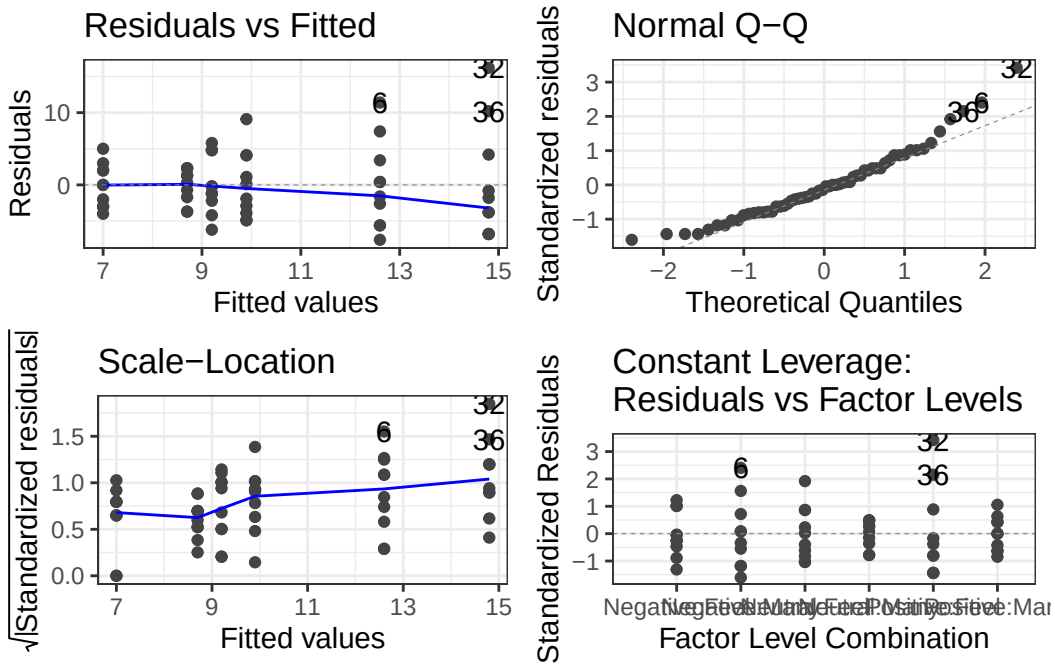
Differs from:

```
 contrast                       estimate   SE df t.ratio p.value p.value
 Positive Feel - Positive Many       7.8 2.23 54  3.490  0.0118
```

15.7 Diagnostics

Residual diagnostics are just like before. Use the `autoplot()` function to gauge how the assumptions are holding.

```
autoplot(ocd.fit)
```



And if you feel as if you need it, use the `leveneTest()` function in the `car` package to get the Brown-Forsythe test (because levene's apparently isn't the default!)

This must be done at the cell means level, and `leveneTest` doesn't play nice with Two-Way ANOVA models. You have to specify a formula with JUST the interaction term.

```
leveneTest(items ~ mood:stopRule, ocd)
```

```
Levene's Test for Homogeneity of Variance (center = median)
      Df F value Pr(>F)
group  5  1.7291 0.1436
      54
```

16 Unbalanced Two-Factor Analysis of Variance

Here are some code chunks that setup this chapter.

```
# Here are the libraries I used
library(tidyverse) # standard
library(knitr)

library(readr) # need to read in data
library(ggpubr) # allows for stat_cor in ggplots
library(ggfortify) # Needed for autoplot to work on lm()
library(gridExtra) # allows me to organize the graphs in a grid
library(car) # need for some regression stuff like vif
library(GGally)
library(emmeans)
```

```
# This changes the default theme of ggplot
old.theme <- theme_get()
theme_set(theme_bw())
```

An **Unbalanced** ANOVA involves data where individual treatments/cells do not have the same number of observations/subjects.

The principals remain the same:

- y is our response variable
- We can have various x variables that we are manipulating.
  - In ANOVA context these are called **factors**.
  - We can have more than one factor. (Technically as many as we want, but probably stop at 3!)
  - Each factor has a set of levels, i.e. the specific values that are set
  - A **treatment** is the specific combination of the levels of all the factors.
- We wish to assess which *factors* are important in our response variable.
- The main diffference with unbalanced data is that the hypothesis tests become a little more obtuse.

16.1 Two way ANOVA Model

This is still the same! It's just copy pasted from previous section.

There is a **means model**:

$$y = \mu_{ij} + \epsilon$$

- μ_{ij} is the mean of treatment A_iB_j.
- Epsilon is the error term and it is assumed $\epsilon \sim N(0, \sigma)$.
  - The constant variability assumption is here.

Which in my opinion is not congruent with what we are trying to investigate in Two-Way ANOVA.

- We are trying to understand the effect that both factors have on the the response variable.
- There may be an **interaction** between the factors.
  - This is when the effect the levels of factor A is not consistent across all levels of factor B, and vice versa.
  - For example, the mean of the response variable may increase when going from treatment A_1B_1 to A_2B_1, but the response variable mean decreases when we look at A_1B_2 to A_2B_2.

With this all in mind, the **effects model** in two-way ANOVA is:

$$y = \mu + \alpha_i + \beta_j + \gamma_{ij} + \epsilon$$

- μ would be the mean of the response variable without the effects of the factors.
  - This can usually be thought of as the mean of a control group.
- α_i can be thought of as the effect that level i of factor A has on the mean of y.
- Likewise, β_j is the effect that level j of factor B has on the mean of y.
- γ_{ij} is the interaction effect, which is the additional effect that the specific combination A_i and B_j has on the mean of y.

16.2 Notation and jargon

There's a lot here. The notation logic is fairly procedural (to a degree that it becomes confusing).

Factors and Treatments

We have factor A with levels $i = 1, 2, \ldots, a$ and factor B with levels $j = 1, 2, \ldots, b$.

- A_i represents level i of factor A.
- B_j represents level j of factor B.
- A_iB_j represents the treatment combination of level i and level j of factors A and B respectively.
- This is sometimes referred to as "treatment ij".

Sample Sizes

This is where things get sketchier/more obsuse. We have to account for the fact that each A_iB_j treatment group may have a different sample size.

- n_{ij} is the number of observations/subjects in treatment A_iB_j.
- $n_{i.}$ represents the total number of observations in A_i. ($n_{i.} = \sum_{j=1}^{b} n_{ij}$)
- $n_{.j}$ represents the total number of observations in level i of factor A. ($n_{.j} = \sum_{i=1}^{a} n_{ij}$)
- $n_{..}$ (or sometimes N) represents the total number of observations overall. ($n_{..} = \sum_{i=1}^{a}\sum_{j=1}^{b} n_{ij}$)

Observations and Sample Means

An individual observation is denoted by y_{ijk}. The subscripts indicate we are looking at observation k from treatment A_iB_j

- Since there are n_{ij} observations in each individual treatment treatment, $k = 1, 2, \ldots, n_{ij}$.
- $\overline{y}_{ij.}$ is the mean of the observations in treatment A_iB_j. ($\overline{y}_{ij.} = \frac{1}{n_{ij}}\sum_{k=1}^{n_{ij}} y_{ijk}$)
- $\overline{y}_{i..}$ is the mean of the observations in treatment A_i across all levels of factor B. ($\overline{y}_{i..} = \frac{1}{n_{i.}}\sum_{j=1}^{b}\sum_{k=1}^{n_{ij}} y_{ijk}$)
- $\overline{y}_{.j.}$ is the mean of the observations in treatment B_j across all levels of factor A. ($\overline{y}_{.j.} = \frac{1}{n_{.j}}\sum_{i=1}^{a}\sum_{k=1}^{n_{ij}} y_{ijk}$)
- $\overline{y}_{...}$ is the mean of all observations. ($\overline{y}_{...} = \frac{1}{n_{..}}\sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{k=1}^{n_{ij}} y_{ijk}$)

16.3 Sums of Squares in Unbalanced Designs

This is where things get tricky in unbalanced designs. We are forgoing formulas at this point because they become somewhat less meaningful.

- In order for you to safely perform these hypothesis tests, you have to consider how you are testing them.
- If we were to follow the pattern of say multiple regression or one-way ANOVA, we could separate the variability explained by each factor.

$$SSA = \text{Variability Explained by Factor A}$$

- SSA would what is used to test $H_0 : \alpha_i = 0$ for all i.

$$SSB = \text{Variability Explained by Factor B}$$

- SSB would what is used to test $H_0 : \beta_j = 0$ for all j.

$$SSAB = \text{Variability Explained by interaction between A and B}$$

- SSAB would what is used to test $H_0 : \gamma_{ij} = 0$ for all i, j.

However… In unbalanced designs, we no longer have access to these simplified sums of squares and hypothesis tests

There become 3 different mainstream types of ANOVA Hypothesis Tests

16.4 The Forsest of Sums of Squares

There are many ways of framing the sums of squares in unbalanced ANOVA as care must be taken.

The notation here is changing to emphasize that we are looking at fundamentally different things now.

- $SS(A)$ is the amount of variability explained by factor A in a model by itself,
- $SS(B)$ is the amount of variability explained for by factor B in a model by itself.
- $SS(AB)$ is the amount of variability explained for by interaction in a model by itself.
- $SS(A|B)$ is the additional variability explained for by factor A when factor B is in the model
  - $SS(A|B, AB)$ is the additional variability explained for by factor A when factor B and the interaction are in the model.
- $SS(B|A)$ is the additional variability explained for by factor B when factor A is in the model.
  - $SS(B|A, AB)$ is the additional variability explained for by factor A when factor B and the interaction are in the model.
- $SS(AB|A, B)$ is the additional variability explained for by the interaction when factor B and A are in the model.

The sum of squares are now about how much *more* a factor or interaction can contribute to a model. Not whether it *is* in the model or not.

16.5 Hypothesis Tests

There are three types of tests that use these sums of squares.

- Type I: "Sequential".
- TYpe II: "Marginality" (This is a hill some people choose to die on it seems.)
- Type III: "???"

16.5.1 Type I Tests

Type I tests are about *sequentially* adding terms, starting with the factors then moving up to higher order terms (interaction).

The sequence of hypothesis tests would be the following.

1. Start with one factor, the test would be:

- $H_0 : \mu_{ij} = \mu$
- $H_1 : \mu_{ij} = \mu + \alpha_i$
- This would be performed using $SS(A)$

2. Add in the next factor, the test would be:

- $H_0 : \mu_{ij} = \mu + \alpha_i$
- $H_1 : \mu_{ij} = \mu + \alpha_i + \beta_j$
- This would be performed using $SS(B|A)$

3. Add in the interaction.

- $H_0 : \mu_{ij} = \mu + \alpha_i + \beta_j$
- $H_1 : \mu_{ij} = \mu + \alpha_i + \beta_j + \gamma_{ij}$
- This would be performed using $SS(AB|A, B)$

16.5.2 Type II Tests

Type II Tests run on the principal of "marginality".

- The idea is to move from lower order models, to higher order models with interactions.
- This becomes more important with three or more factors.

1. Compare how well factor A improves a model with factor B in it.

- $H_0 : \mu_{ij} = \mu + \beta_j$
- $H_1 : \mu_{ij} = \mu + \alpha_i + \beta_j$
- This would be performed using $SS(A)$

2. Compare how well factor B improves a model with factor A in it.

- $H_0 : \mu_{ij} = \mu + \alpha_i$
- $H_1 : \mu_{ij} = \mu + \alpha_i + \beta_j$
- This would be performed using $SS(B|A)$

3. Finally, see how well adding in the interaction works when both main effects are in the model:

- $H_0 : \mu_{ij} = \mu + \alpha_i + \beta_j$
- $H_1 : \mu_{ij} = \mu + \alpha_i + \beta_j + \gamma_{ij}$
- This would be performed using $SS(AB|A, B)$

16.5.3 Type III Tests

Type III tests are a kind of catch all test the compares a full model (all terms including interaction) to a model missing a single term.

1. Compare how well factor A improves a model with factor B and the interaction in it.

- $H_0 : \mu_{ij} = \mu + \beta_j + \gamma_{ij}$
- $H_1 : \mu_{ij} = \mu + \alpha_i + \beta_j + \gamma_{ij}$
- This would be performed using $SS(A|B, AB)$

2. Compare how well factor B improves a model with factor A and the interaction in it.

- $H_0 : \mu_{ij} = \mu + \alpha_i + \gamma_{ij}$
- $H_1 : \mu_{ij} = \mu + \alpha_i + \beta_j + \gamma_{ij}$
- This would be performed using $SS(B|A, AB)$

3. Finally, see how well adding in the interaction works when both main effects are in the model:

- $H_0 : \mu_{ij} = \mu + \alpha_i + \beta_j$
- $H_1 : \mu_{ij} = \mu + \alpha_i + \beta_j + \gamma_{ij}$
- This would be performed using $SS(AB|A, B)$

16.5.4 Which tests to use?

In simple terms, it is situationally dependent which hypothesis test you use.

In more practical terms, pretty much Type II and Type III only

- Type I should only be done if you have pre-determined the list of importance of variables, for some reason or another.
- Type II involves a more elegant approach that only becomes important in a three-way ANOVA (3 factors)
  - The general idea is that if a factor is "insignificant" then higher order terms (interactions) would/should not be in the model.
  - Type II sums of squares are most powerful for detecting main effects when interactions are not present.
- Type III is just brute force your way through the hypothesis tests.
- I see Type II recommended more often than Type III. Go ahead and use it by default, I'd say.
- But if you do not really know what is going on Type III is "safe", in my opinion.
  - Others would argue otherwise simply by fact that another option exists and the whole "my opinion is better" mindset.
  - I personally don't see a problem with Type III but it's situation dependent.
  - If there are interactions, then it doesn't even matter because you can't ignore lower order terms anyway.
  - This guy has a VERY strong opinion on not using Type III: Page 12 of https://www.stats.ox.ac.uk/pub/MASS3/Exegeses.pdf.

16.5.5 PAY ATTENTION TO WHAT THE SOFTWARE DOES

- In unmodified/base `R`, an ANOVA analysis will use Type I Tests which is almost NEVER recommended.
- The `Anova()` function in the `car` package uses Type II by defaul, which is fine.
- If you want Type III sum of squares, you have to change an option in `R`, then use the `Anova()` function.
- Other software use different Type tests by default. Read the manual.

16.6 Patient Satisifcation Data

This data is taken from the text: Applied Regression Analysis and Other Multivariable Methods

ISBN: 9781285051086

It's a good book, but it may dive a bit too much in to theory sometimes. I don't feel like I am shamelessly ripping from the book since they took the data from a study/dissertation:

Thompson, S.J. 1972. "The Doctor–Patient Relationship and Outcomes of Pregnancy." Ph.D. disser-tation, Department of Epidemiology, University of North Carolina, Chapel Hill, N.C.

The study attempted to ascertain a patients satisfaction with level of medical care during pregnancy, and its association with how worried the patient was and how affective patient-doctor communication was rated.

Data are available in `patSas.csv`.

- `Satisfaction` is the patient's self rated satisfaction with their medical care.
  - The scale is 1 through 10.
  - It is unknown whether lower is better.
  - It could be a Very Satisfied to Very Dissatisfied from left to right which would mean 1 = Very Satisfied.
- `Worry` is how worried a patient was.
  - `Worry` was grouped into the levels `Negative` and `Postive`.
  - I cannot ascertain whether Negative means they had low worry or they a negative feelings. The book source does not specify, and the dissertation is not open access.
- `AffCom` is the rating of how affective the patient-doctor communication was rated.
  - The levels are `High`, `Medium`, to `Low`.
  - It *may* seem more obvious what the meaning is here, but always be careful. PhD students can be quirky and have their own unique concepts of perceived reality.

16.6.1 Looking at the sample sizes for the cells

```
patSas <- read_csv(here::here(
  "datasets",'patSas.csv'))
```

Here is what a sample of the data look like

```
# A tibble: 10 x 3
   Satisfaction Worry    AffCom
          <dbl> <chr>    <chr>
 1            4 Positive High
 2            2 Negative High
 3            5 Positive Low
 4            6 Positive High
 5            6 Negative High
 6            4 Negative High
 7            4 Positive High
 8            6 Positive High
 9            9 Negative Low
10            7 Positive Medium
```

Lets look at how many patients fall into each factor level and treatment.

```
# Worriers breakdown
table(patSas$Worry)
```

```
Negative Positive
      22       30
```

```
# Communication breakdown
table(patSas$AffCom)
```

```
  High    Low Medium
    22     18     12
```

```r
# Two-way Table
table(patSas$Worry, patSas$AffCom)
```

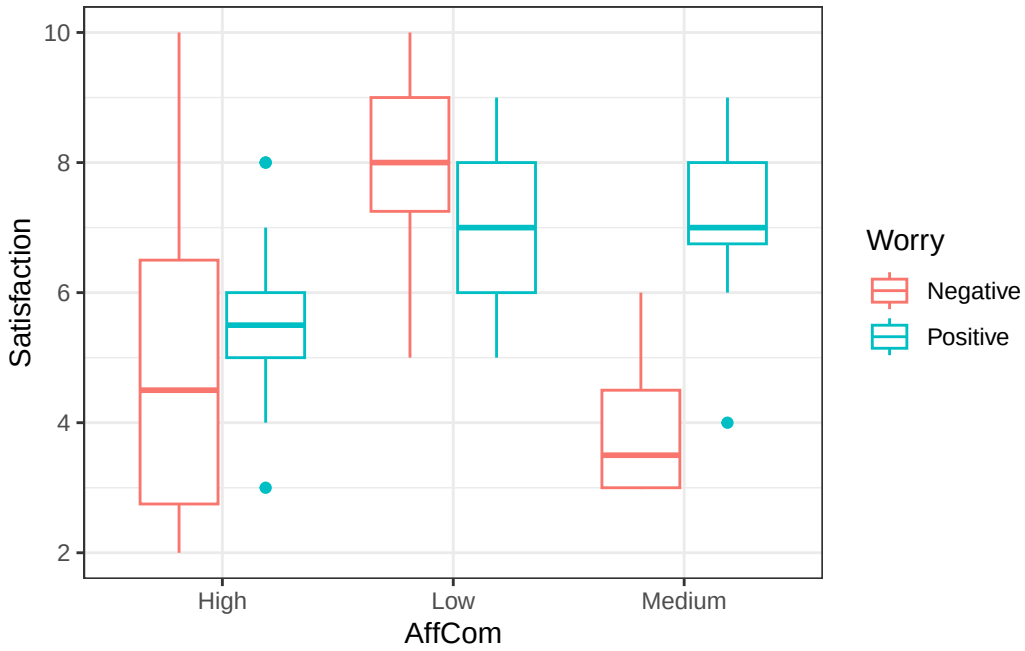
```
          High Low Medium
  Negative    8  10      4
  Positive   14   8      8
```

The setup is unbalanced.

- The main issue I have is the sample size in the Negative/Medium treatment cell is low compared to the others.
- If the ratio between smaller and larger groups gets too large, ANOVA starts becoming kind of pointless.
- a 4 to 14 is not terrible but definitely it is starting to get to the point where ANOVA may not be very useful.

16.6.2 Graphing the data! (DO IT!)

```
ggplot(patSas, aes(y = Satisfaction,
                   x = AffCom, color = Worry)) +
  geom_boxplot()
```



Looks like an interaction. The pattern is different in the negative group versus the positive group.

16.6.3 Type II ANOVA

First, lets use the default of the `Anova()` function in the `car` package.

This will do Type II Tests by default

```
# Make your model
pat.fit <- lm(Satisfaction ~ AffCom*Worry, patSas)

Anova(pat.fit)
```

```
Anova Table (Type II tests)

Response: Satisfaction
              Sum Sq Df F value    Pr(>F)
AffCom        51.829  2  8.3112 0.0008292 ***
Worry          3.425  1  1.0984 0.3000824
AffCom:Worry  26.682  2  4.2787 0.0197611 *
Residuals    143.429 46
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Here we have somewhat strong evidence that there is an interaction.

- This wouldn't fall under my default criteria for declaring "significance" if I were in the NHST frame of reference.
- Interaction is fairly apparent from the graphs.
- I would err on the side of caution, and declare that there is one.
  - We would have to ignore main effects, but if there is actually an interaction, main effect analyses would be difficult to interpret.
  - It *might* be prudent for higher α values for interaction tests in general.

16.6.4 Type III ANOVA

To do type III ANOVA in R, specifically within the `Anova()` function, you have to do a couple things.

- There is a `type` argument so you would use `Anova(model, type = "III")` or `type = 3`.
- You need also need to specify a special option.
- Then you have to create a new model AFTER you use that option.
- This is usually the default method in most statistical software (other than `car` in R)

```
options(contrasts = c("contr.sum","contr.poly"))

pat.fit2 <- lm(Satisfaction ~ AffCom*Worry, patSas)

Anova(pat.fit2, type = 3)
```

```
Anova Table (Type III tests)

Response: Satisfaction
              Sum Sq Df  F value    Pr(>F)
(Intercept)  1679.33  1 538.5911 < 2.2e-16 ***
AffCom         52.11  2   8.3566  0.000802 ***
Worry           8.30  1   2.6627  0.109554
AffCom:Worry   26.68  2   4.2787  0.019761 *
Residuals     143.43 46
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Notice that intercept term there, ignore it. That's just telling us there is an overall mean that is not zero, which is a pointless piece of information given the scores start at 1.

16.6.5 Multiple Comparisons

Everything is the same as previously when using `emmeans()` and `pairs()`.

- If there is an interaction, then you must use `emmeans(model, ~ A:B)` or `A*B`.
- When using `pairs` it is safer to use the `adjust = "holm"` argument.
  - Tukey procedures get more unreliable in the case of unbalanced data.
  - This procedure conserves FWER under arbitrary conditions and is better than Bonferroni.

16.6.6 Main Effect Comparisons

If we consider the evidence of interaction to be sufficient to conclude that there is in fact an interaction, it is arguably useless to look at main effects.

- Furthermore, we would not look at the Worry main effect means and pairwise comparisons.
- This is just for demonstration.

```
worryMeans <- emmeans(pat.fit, ~ Worry)

worryMeans
```

```
 Worry    emmean    SE df lower.CL upper.CL
 Negative   5.67 0.406 46     4.85     6.48
 Positive   6.52 0.334 46     5.85     7.20

Results are averaged over the levels of: AffCom
Confidence level used: 0.95
```

```
pairs(worryMeans)
```

```
 contrast            estimate    SE df t.ratio p.value
 Negative - Positive   -0.857 0.525 46  -1.632  0.1096

Results are averaged over the levels of: AffCom
```

- There still was an interaction so this comparison may not be considered worthwhile.

```r
comMeans <- emmeans(pat.fit, ~ AffCom)

comMeans
```

```
 AffCom emmean    SE df lower.CL upper.CL
 High     5.29 0.391 46     4.50     6.07
 Low      7.50 0.419 46     6.66     8.34
 Medium   5.50 0.541 46     4.41     6.59

Results are averaged over the levels of: Worry
Confidence level used: 0.95
```

```r
pairs(comMeans, adjust = "holm")
```

```
 contrast      estimate    SE df t.ratio p.value
 High - Low      -2.214 0.573 46  -3.863  0.0010
 High - Medium   -0.214 0.667 46  -0.321  0.7496
 Low - Medium     2.000 0.684 46   2.924  0.0107

Results are averaged over the levels of: Worry
P value adjustment: holm method for 3 tests
```

This indicates that high and medium affective communication were associated with lower patient satisfaction scores.

16.6.7 One Way to Look at Interactions: AffCom by Worry Levels

We may only be interested in how affective communication affects satisfaction when looking at the individual levels of worry.

- This is a comparison of cell/interaction means.
- However, we are subsetting the comparisons we care about.

```
comByWorryMeans <- emmeans(pat.fit, ~ AffCom | Worry)

comByWorryMeans
```

```
Worry = Negative:
 AffCom emmean    SE df lower.CL upper.CL
 High     5.00 0.624 46     3.74     6.26
 Low      8.00 0.558 46     6.88     9.12
 Medium   4.00 0.883 46     2.22     5.78

Worry = Positive:
 AffCom emmean    SE df lower.CL upper.CL
 High     5.57 0.472 46     4.62     6.52
 Low      7.00 0.624 46     5.74     8.26
 Medium   7.00 0.624 46     5.74     8.26

Confidence level used: 0.95
```

```
pairs(comByWorryMeans, adjust = "holm")
```

```
Worry = Negative:
 contrast      estimate    SE df t.ratio p.value
 High - Low       -3.00 0.838 46  -3.582  0.0016
 High - Medium     1.00 1.080 46   0.925  0.3599
 Low - Medium      4.00 1.040 46   3.829  0.0012

Worry = Positive:
 contrast      estimate    SE df t.ratio p.value
 High - Low       -1.43 0.783 46  -1.825  0.2233
 High - Medium    -1.43 0.783 46  -1.825  0.2233
 Low - Medium      0.00 0.883 46   0.000  1.0000

P value adjustment: holm method for 3 tests
```

- So it seems that the discrepancy arises when worry levels are categorized as “Negative”
- However, even though there is low significance, the same pattern arises in the Worry = Positive group.

16.6.8 Or Maybe: Worry by AffCom Levels

We may only be interested in how affective communication affects satisfaction when looking at the individual levels of worry.

- This is a comparison of cell/interaction means.
- However, we are subsetting the comparisons we care about.

```
worryByComMeans <- emmeans(pat.fit, ~ Worry | AffCom)

worryByComMeans
```

```
AffCom = High:
 Worry    emmean    SE df lower.CL upper.CL
 Negative   5.00 0.624 46     3.74     6.26
 Positive   5.57 0.472 46     4.62     6.52

AffCom = Low:
 Worry    emmean    SE df lower.CL upper.CL
 Negative   8.00 0.558 46     6.88     9.12
 Positive   7.00 0.624 46     5.74     8.26

AffCom = Medium:
 Worry    emmean    SE df lower.CL upper.CL
 Negative   4.00 0.883 46     2.22     5.78
 Positive   7.00 0.624 46     5.74     8.26

Confidence level used: 0.95
```

```
pairs(worryByComMeans, adjust = "holm")
```

```
AffCom = High:
 contrast            estimate    SE df t.ratio p.value
 Negative - Positive   -0.571 0.783 46  -0.730  0.4690

AffCom = Low:
 contrast            estimate    SE df t.ratio p.value
 Negative - Positive    1.000 0.838 46   1.194  0.2386

AffCom = Medium:
 contrast            estimate    SE df t.ratio p.value
 Negative - Positive   -3.000 1.080 46  -2.774  0.0080
```

16.6.9 Another way: All Pairwise Comparison (Throw everything at the wall and see what sticks)

```
cellMeans <- emmeans(pat.fit, ~ AffCom:Worry)

cellMeans
```

```
 AffCom Worry    emmean    SE df lower.CL upper.CL
 High   Negative   5.00 0.624 46     3.74     6.26
 Low    Negative   8.00 0.558 46     6.88     9.12
 Medium Negative   4.00 0.883 46     2.22     5.78
 High   Positive   5.57 0.472 46     4.62     6.52
 Low    Positive   7.00 0.624 46     5.74     8.26
 Medium Positive   7.00 0.624 46     5.74     8.26

Confidence level used: 0.95
```

```
pairs(cellMeans, adjust = "holm")
```

```
 contrast                          estimate    SE df t.ratio p.value
 High Negative - Low Negative        -3.000 0.838 46  -3.582  0.0115
 High Negative - Medium Negative      1.000 1.080 46   0.925  1.0000
 High Negative - High Positive       -0.571 0.783 46  -0.730  1.0000
 High Negative - Low Positive        -2.000 0.883 46  -2.265  0.2825
 High Negative - Medium Positive     -2.000 0.883 46  -2.265  0.2825
 Low Negative - Medium Negative       4.000 1.040 46   3.829  0.0058
 Low Negative - High Positive         2.429 0.731 46   3.322  0.0229
 Low Negative - Low Positive          1.000 0.838 46   1.194  1.0000
 Low Negative - Medium Positive       1.000 0.838 46   1.194  1.0000
 Medium Negative - High Positive     -1.571 1.000 46  -1.570  0.7400
 Medium Negative - Low Positive      -3.000 1.080 46  -2.774  0.0956
 Medium Negative - Medium Positive   -3.000 1.080 46  -2.774  0.0956
 High Positive - Low Positive        -1.429 0.783 46  -1.825  0.5955
 High Positive - Medium Positive     -1.429 0.783 46  -1.825  0.5955
 Low Positive - Medium Positive       0.000 0.883 46   0.000  1.0000

P value adjustment: holm method for 15 tests
```

And then you have this mess to summarize. (Cell references are in AffCom:Worry format)

- Low:Negative have higher mean scores than High:Negative, High:Positive, and Medium:Negative. ($p \leq 0.0229$)
- All other comparisons are inconclusive.

A generalization of this is that Low affective communication and Negative worry were associated with higher satisfaction scores.

- Hopefully this means that the scale goes from very satisfied to very disatisfied in terms of 1 through 10 and affective communication leads to better satisfaction.
- Otherwise if a patient is categorized and "negative" on the worry skill, we should send in a doctor that sucks at communicating, apparently.

17 Generalized Linear Models

```
# Need the knitr package to set chunk options
library(knitr)
library(tidyverse)
library(gridExtra)
library(caret)
library(readr)
```

It should be noted that depending on the time frame that you may read about "GLMs", this may refer to two different types of modeling schemes

- GLM → General Linear Model: This is the older scheme that now refers to Linear Mixed Models or LMMs. This has more with correlated error terms and "random effects" in model.
- These days GLM refers to **Generalized Linear Model**. Developed by Nelder, John; Wedderburn, Robert (1972). This takes the concept of a linear model and generalizes it to response variables that do not have a normal distribution. **This is what GLM means today**. (I haven't run into an exception, but there might be ones out there depending on the discipline.)

17.1 Components of Linear Model

We will start with the idea of the regression model we have been working.

$$\mu_{y|x} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p.$$

- Here we are saying that the mean value of y, $\mu_{y|x}$, is determined by the values of our predictor variables.
- For making predictions, we have to add in a random component (because no model is perfect).

$$y = \mu_{y|x} + \epsilon = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + \epsilon.$$

We have two components here.

1. A deterministic component.
2. A random component: $\epsilon \sim N(0, \sigma^2)$.

- The random component means that $Y \sim N(\mu_{y|x}, \sigma^2)$.
- The important part here is that the mean value of y is assumed to be *linearly* related to our x variables.

17.1.1 Introduction to Link Functions

We have tried **transformations** on data to try to see if we can fix heterogenity and non-linearity issues.

- In Generalized Linear Models these are known as **link functions**.
- The functions are meant to *link* a linear model to the distribution of y or link the distribution of y to a linear model. (which ever way you want to think about it)
- In the case of linear regression they are supposed to be the **link** of y with Normality and *linearity*.

Example:

$$g(y) = \mathsf{log}(y) \quad = \eta_{y|x} + \epsilon = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + \epsilon$$

What does this mean for y itself?

$$Y = g^{-1}\left(\eta_{y|x} + \epsilon\right) = e^{\eta_{y|x}+\epsilon}$$
$$\mu_{y|x} = e^{\beta_0+\beta_1 x_1+\beta_2 x_2+\cdots+\beta_p x_p+\epsilon}$$

- Another common option is $g(Y) = Y^p$ where it could be hoped that $p = 1$ but more often the best p is not…

17.1.2 Form of Generalized Linear Models

The form of Generalized Linear Models takes the basic form:

- Observations of y are the result of some probability distribution whose mean (and potentially variance) are determined or correlated with certain "predictor"/x variables.
- The mean value of y is $\mu_{y|x}$ and is not linearly related to x, but via the **link function** it is, i.e.,

$$g(\mu_{y|x}) =$$

There are now three components we consider:

1. A deterministic component.
2. The probability distribution of y with some mean $\mu_{y|x}$ and standard deviation (randomness) $\sigma_{y|x}$.

- In ordinary linear regression the probability distribution for y was

$$N(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p, \sigma)$$

- Consider y from a binomial distribution with probability of success $p(x)$, that is, the probability depends on x. For an individual trial, we have:
  - $\mu_{y|x} = p(x)$
  - $\sigma_{y|x} = \sqrt{p(x)(1-p(x))}$
  - This makes is so we can no longer just chuck the error term into the linear equation for the mean to represent the randomness.

3. A link function $g(y)$ such that

$$\begin{aligned} g\left(\mu_{y|x}\right) &= \eta_{y|x} \\ &= \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p \end{aligned}$$

With GLMs, we are now paying attention to a few details.

- y is not restricted to the normal distributions; that's the entire point…
- The link function is chosen based on the distribution of y
  - The distribution of y determines the form of $\mu_{y|x}$.

17.1.3 Various Types of GLMs

Distribution of Y	**Support of Y**	**Typical Uses**	**Link Name**	**Link Function: $g(y)$**
Normal	$(-\infty, \infty)$	Linear response data	Identity	$g\left(\mu_{y\|x}\right) = \mu_{y\|x}$
Exponentail	$(0, \infty)$	Exponential Processes	Inverse	$g\left(\mu_{y\|x}\right) = \frac{1}{\mu_{y\|x}}$
Poisson	$0, 1, 2, ...$	Counting	Log	$g\left(\mu_{y\|x}\right) = \log_{y\|x}$
Binomial	$0, 1$	Classification	Logit	$g\left(\mu_{y\|x}\right) = \ln\left(\frac{\mu_{y\|x}}{1-\mu_{y\|x}}\right)$

We will mainly explore Logistic Regression, which uses the Logit function.

This is one of the ways of dealing with Classification.

17.2 Classification Problems, In General.

Classification can be simply define as determing what the outcome is for a discrete random variable.

- An online banking service must be able to determine whether or not a transaction being performed on the site is fraudulent, on the basis of the user's past tranaction history, balance, an other potential predictors. This a common use of statistical classification known as **Fraud detection**
- On the basis of DNA sequence data for a number of patients with and without a given disease a biologist would like to figure out which DNA mutations are disease-causing and which are not.
- A person arrives at the emergency room with a set of symptoms that could possiblly be attributed to one of three medical conditions: *stroke, drug overdose, epileptic seizure.* We would want to choose or **classify** the person into one of the three categories.

In each of these situations what is the distribution of $Y =$ the category an indidual falls into?

17.2.1 Odds and log-odds

In classification models, the idea to to estimate the probability of an event occuring for an individual: p.

There are a couple of ways that we assess how likely an event is:

- Odds are $Odds(p) = \frac{p}{1-p}$
  - This is the ratio of an event occurring versus it not occurring.
  - This gives a numerical estimate of how many times more likely it is for the event to occur rather than not.
  - $Odds = 1$ implies a fifty/fifty split: $p = 0.5$. (Like a coin flip!)
  - $Odds = d$ where $d > 1$ implies that the event is d times more likely to happen than not.
  - For $d < 0$ the event is $1/d$ times more likely to NOT happen rather than happen.
- log-odds are taking the logarithm (typically the natural logarithm) of the odds: $\mathsf{log}(Odds(p)) = log\left(\frac{p}{1-p}\right)\$$
  - In logistic regression, the linear relationship between $p(x)$ and the predictor variables is assumed to be via $\mathsf{log}(Odds)$.
  - $log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k$

17.3 An Example, Default

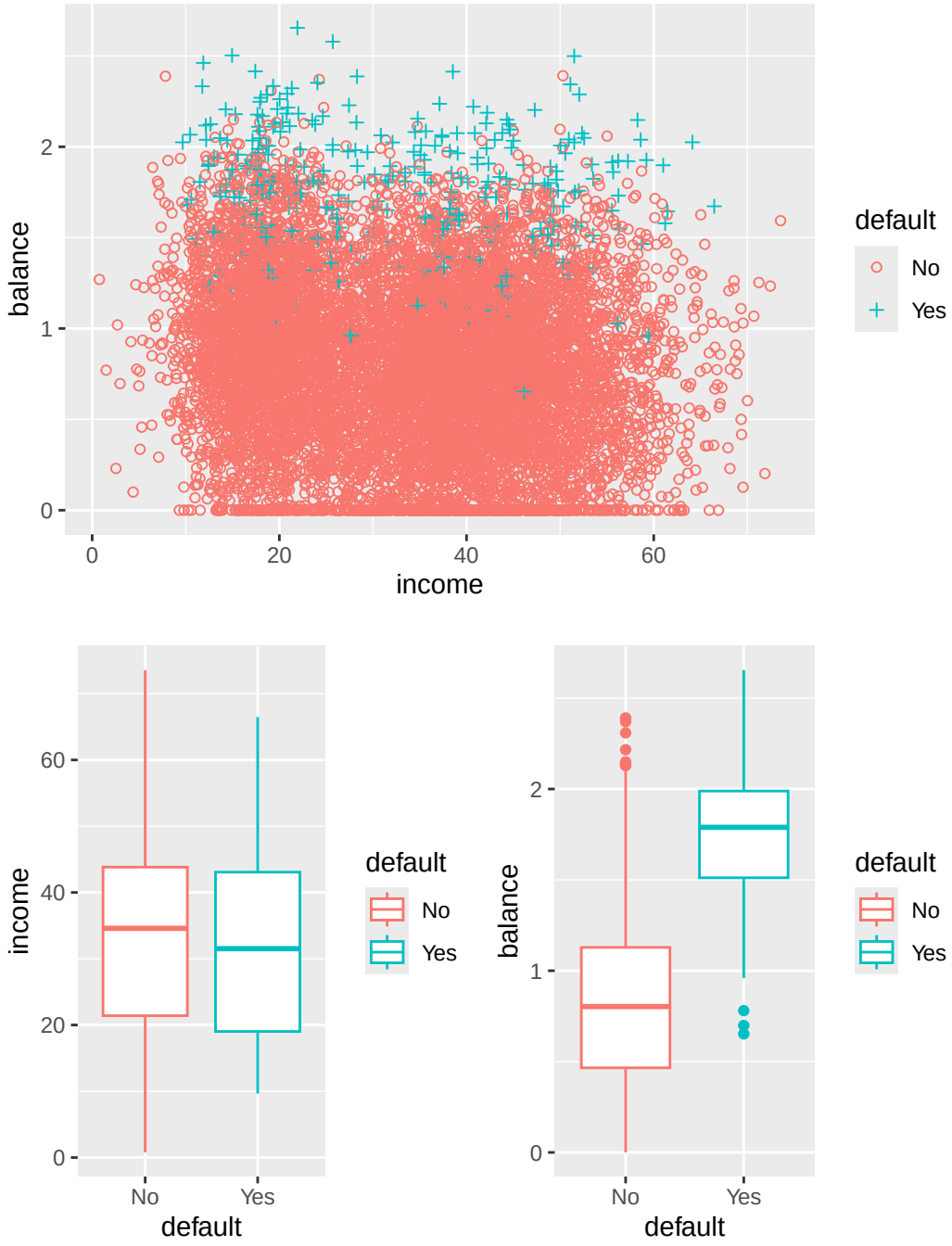
We will consider an example where we are interested in predicting whether an individual will default on their credit card payment, on the basis of annual income, monthly credit card balance and student status. (Did I pay my bills… Maybe I should check.)

```
Default <- read_csv(here::here("datasets","Default.csv"))

# Converting Balance and Income to Thousands of dollars

Default$balance <- Default$balance/1000
Default$income <- Default$income/1000

head(Default)
```

```
# A tibble: 6 x 5
   ...1 default student balance income
  <dbl> <chr>   <chr>     <dbl>  <dbl>
1     1 No      No        0.730  44.4
2     2 No      Yes       0.817  12.1
3     3 No      No        1.07   31.8
4     4 No      No        0.529  35.7
5     5 No      No        0.786  38.5
6     6 No      Yes       0.920   7.49
```

- `student` whether the person is a student or not.
- `default` is whether they defaulted
- `balance` is there total balance on their credit cards.
- `income` is the income of the individual.

17.3.1 A little bit of EDA



Try to describe what you see.

17.3.2 Logistic Regression

For each individual y_i we are trying to model if that individual is going to default or not.

- We will start by modeling $P(\text{Default} \mid \text{Balance}) \equiv P(Y|x)$.
- First, we we will code the default to a 1, 0 format to make modeling this probability compatible with the standard form of the with a special case of the Binomial(n, p) distribution where $n = 1$

$$P(Y = 1) = p(x)$$

- Logistic regression creates a model for $P(Y = 1|x)$ (technically the log-odds thereof) which we will abbreviate as $p(x)$

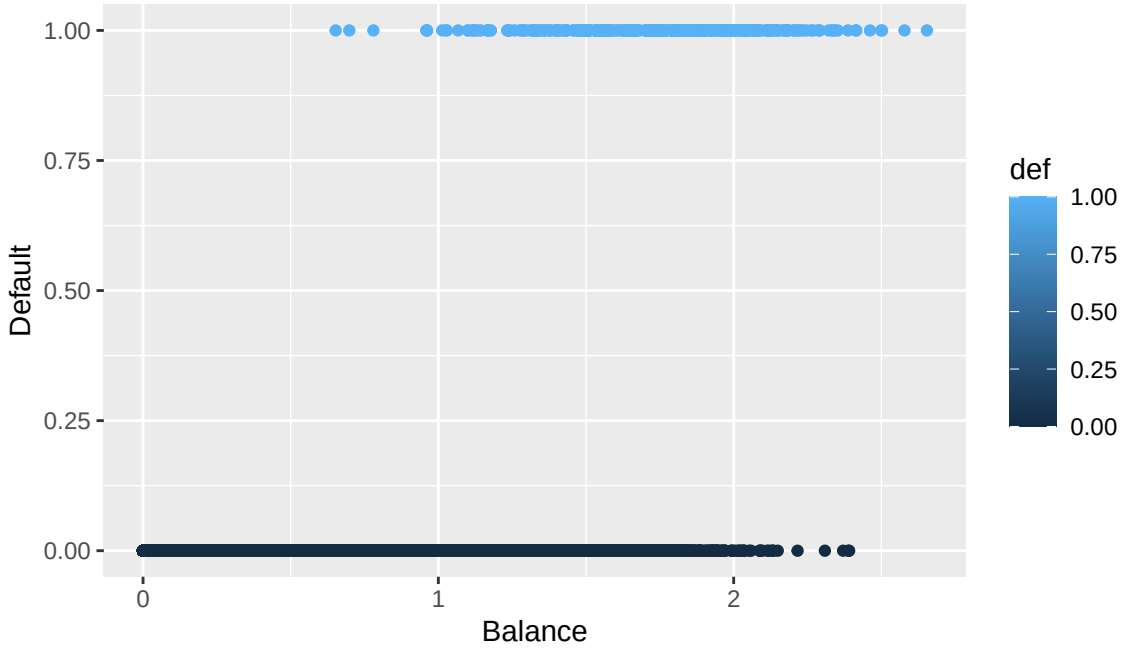
```
# Set an ifelse statement to handle the variable coding

#Create a new varible called def with the coded values
Default$def <- ifelse(Default$default == "Yes", 1,0)
```

Before we get into the actual Logistic Regression model, lets start with the idea that if we are trying to predict $p(x)$, when should we should we say that someone is going to default?

- A possiblity may be we classify the person as potentially defaulting if $p(x) > 0.5$
- Would it make sense to predict somone as a risk for defaulting at some other cutoff?

17.3.3 Plotting

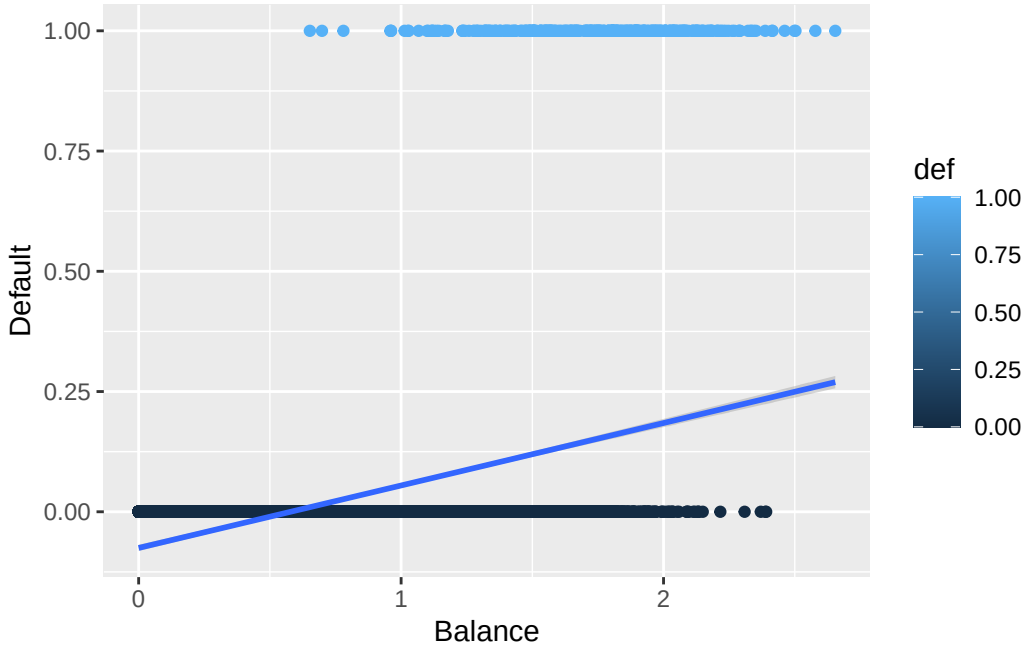
```
def.plot <- ggplot(Default, aes(x = balance, y = def, color = def)) + geom_point() +
  xlab("Balance") + ylab("Default")

def.plot
```



17.3.4 Why Not Linear Regression

We *could* use standard linear regression to model $p(\text{Balance}) = p(x)$. The line on this plot represents the estimated probability of defaulting.

```
def.plot.lm <- ggplot(Default, aes(x = balance, y = def, color = def)) + geom_point()+
  xlab("Balance") + ylab("Default") +
  geom_smooth(method = 'lm')

def.plot.lm
```

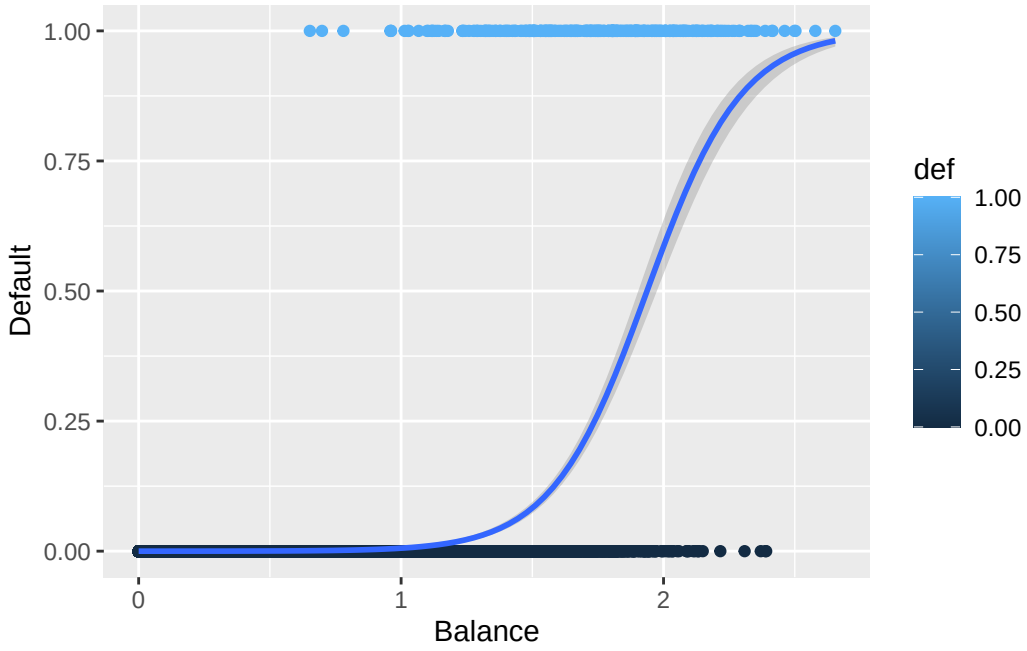


Can you think of any issues with this? Think about possible values of $p(x)$.

17.3.5 Plotting the Logistic Regression Curve

Here is the curve for a logistic regression.

```
def.plot.logit <- ggplot(Default, aes(x = balance, y = def, color = def)) + geom_point() +
  xlab("Balance") + ylab("Default") +
  geom_smooth(method = "glm", method.args = list(family = "binomial"))

def.plot.logit
```



How does this compare to the previous plot?

17.4 The Logistic Model in GLM

Even though we are trying to model a $p(x)$, we are trying to model a mean.

What is the mean or Expected Values of a binomial random variable $Y|x \sim \text{Binomial}(1, p(x))$?

Before we get into the link function, lets look at the function that models $p(x)$

$$p(x) = \frac{e^{\beta_0+\beta_1 x}}{1 + e^{\beta_0+\beta_1 x}}$$

The **link functions** for Logistic Regression is the **logit function** which is

$$\text{logit}(x) = \log\left(\frac{m}{1-m}\right) \text{ where } 0 < x < 1$$

Which for $\mu_{y|x} = p(x)$ is

$$\log\left(\frac{p(x)}{1-p(x)}\right) = \beta_0 + \beta_1 x$$

- Notice that now we have a linear function of the coefficients.
- Something to pay attention to: When we are interpreting the coefficients, we are talking about how the **log odds** change.
- The coefficients tell us what the percentage increase of the odds ratio would be.

$$\text{Odds} = \frac{p(x)}{1-p(x)}$$

17.4.1 Estimating The Coefficients: `glm` function

The function that is used for GLMs in R is the `glm` function.

`glm(formula, family = gaussian, data)`

- `formula`: the linear formula you are using when the link function is applied to $\mu(vx)$. This has same format as `lm`, e.g, `y ~ x`
- `family`: the distribution of Y which in logistic regression is `binomial`
- `data`: the dataframe... as has been the case before.

17.4.2 GLM Function on Default Data

```
default.fit <- glm(def ~ balance, family = binomial, data = Default)

summary(default.fit)
```

```
Call:
glm(formula = def ~ balance, family = binomial, data = Default)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -10.6513     0.3612  -29.49   <2e-16 ***
balance       5.4989     0.2204   24.95   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2920.6  on 9999  degrees of freedom
Residual deviance: 1596.5  on 9998  degrees of freedom
AIC: 1600.5

Number of Fisher Scoring iterations: 8
```

17.4.3 Default Predictions

"Predictions" in GLM models get a bit trickier than previously.

- We can predict in terms of the linear response, $g(\mu(x))$, e.g., log odds in logistic regression.
- Or we can predict in terms of the actual response variable, e.g., $p(x)$ in logistic regression.

$$\log\left(\frac{\hat{p}(x)}{1-\hat{p}(x)}\right) = -10.6513 + 5.4090x$$

Which may not be very useful in application.

log

So for someone with a credict balance of 1000 dollars, we would predict that their probability of defaulting is

$$\hat{p}(x) = \frac{e^{-10.6513+5.4090\cdot 1}}{1+e^{-10.6513+5.4090\cdot 1}} = 0.00526$$

And for someone with a credit balance of 2000 dollars, our prediction probability for defaulting is

$$\hat{p}(x) = \frac{e^{-10.6513+5.4090\cdot 2}}{1+e^{-10.6513+5.4090\cdot 2}} = 0.54158$$

17.5 Interpreting coefficients

Because of what's going on in logistic regression, we have forms for our model:

1. The linear form which is for predicting log-odds.
2. Odds ratios which is the exponentiated form of the model, i.e., $e^{\beta_0+\beta_1 x}$.

- Then, we look at odds increasing (positive β_1) or decreasing (negative β_1) by e^{β_1} times.

3. Probabilities which are of the form $\frac{e^{\beta_0+\beta_1 x}}{1+e^{\beta_0+\beta_1 x}}$.

- Then β_1 means… Nothing really intuitive.
- You just say β_1 indicates increasing or decreasing probability respective of whether β_1 is postive or negative.

Honestly, GLMs are where interpretations get left behind because the link functions screw with our ability to make sense of the math.

17.6 Predict Function for GLMS

We can compute our predictions by hand but, that's not super efficient. We can infact use the `predict` function on `glm` objects just like we did with `lm` objects.

With GLMs, our `predict` function now takes the form:

```
predict(glm.model, newdata, type)
```

- `glm.model` is the `glm` object you created to model the data.
- `newdata` is the data frame with values of your predictor variables you want predictions for. If no argument is given (or its incorrectly formatted) you will get the fitted values for your training data.
- `type` chooses which form of prediction you want.
  - `type = "link"` (the default option) gives the predictions for the link function, i.e, the linear function of the GLM. So for logistic regression it will spit out the predicted values of the log odds.
  - `type = "response"` gives the prediction in terms of your response variable. For Logistic Regression, this is the predicted probabilities.
  - `type = "terms"` returns a matrix giving the fitted values of each term in the model formula on the linear predictor scale. (idk)

17.6.1 Using `predict` on Default Data

Let's get predictions from the logistic regression model that's been created.

```
predict(default.fit, newdata = data.frame(balance = c(1,2)), type = "link")
```

```
         1          2
-5.1524137  0.3465032
```

```
predict(default.fit, newdata = data.frame(balance = c(1,2)), type = "response")
```

```
          1           2
0.005752145 0.585769370
```

17.6.2 Classifying Predictions

If we are looking at the log odds ratio, we will classify an observation as a $\widehat{Y} = 1$ if the log odds is non-negative.

```
Default$pred.link <- predict(default.fit)
Default$pred.class1 <-as.factor(ifelse(Default$pred.link >=0, 1, 0))
```

Or we can classify based off the predicted probabilities, i.e., $\widehat{Y} = 1$ if $\hat{p}(x) \geq 0.5$.

```
Default$pred.response <- predict(default.fit, type = "response")
Default$pred.class2 <-as.factor(ifelse(Default$pred.response >=0.5, 1, 0))
```

Which will produce identical classifications.

```
sum(Default$pred.class1 != Default$pred.class2)
```

```
[1] 0
```

17.6.3 Assessing Model Accuracy

We can assess the accuracy of the model using a confusion matrix using the `confusionMatrix` function, which is part of the `caret` package.

- Note that this will require you to install the `e1071` package (which I can't fathom why they would do it this way…).

```
confusionMatrix(Predicted, Actual)
```

```
confusionMatrix(Default$pred.class1, as.factor(Default$def))
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 9625  233
         1   42  100

               Accuracy : 0.9725
                 95% CI : (0.9691, 0.9756)
    No Information Rate : 0.9667
    P-Value [Acc > NIR] : 0.0004973

                  Kappa : 0.4093

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9957
            Specificity : 0.3003
         Pos Pred Value : 0.9764
         Neg Pred Value : 0.7042
             Prevalence : 0.9667
         Detection Rate : 0.9625
   Detection Prevalence : 0.9858
      Balanced Accuracy : 0.6480

       'Positive' Class : 0
```

17.7 Categorical Predictors

Having categorical variables and including multiple predictors is pretty easy. It's just like with standard linear regression.

Let's look at the model that just uses the `student` variable to predict the probability of defaulting.

```
default.fit2 <- glm(def ~ student,
                    family = binomial,
                    data = Default)

summary(default.fit2)
```

```

Call:
glm(formula = def ~ student, family = binomial, data = Default)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.50413    0.07071  -49.55  < 2e-16 ***
studentYes   0.40489    0.11502    3.52 0.000431 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2920.6  on 9999  degrees of freedom
Residual deviance: 2908.7  on 9998  degrees of freedom
AIC: 2912.7

Number of Fisher Scoring iterations: 6
```

```
predict(default.fit2,
        newdata = data.frame(student = c("Yes",
                                         "No")),
        type = "response")
```

```
         1          2
0.04313859 0.02919501
```

Would student status by itself be useful for classifying individuals as defaulting or no?

17.8 Multiple Predictors

Similarly (Why isn't there an 'i' in that?) to categorical predictors we can easily include multiple predictor variables. Why? Because with the link function, we are just doing linear regression.

$$p_{y|x} = \frac{e^{\beta_0+\beta_1 x_1+\beta_2 x_2 \cdots+\beta_p x_p}}{1+e^{\beta_0+\beta_1 x_1+\beta_2 x_2 \cdots+\beta_p x_p}}$$

The **link functions** is still the same.

$$\text{logit}(m) = \log\left(\frac{m}{1-m}\right) \text{ where } 0 < x < 1$$

Which for $\mu_{y|x} = p_{y|x}$ is

$$\log\left(\frac{p_{y|x}}{1-p_{y|x}}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 \cdots + \beta_p x_p$$

17.8.1 Multiple Predictors in Default Data

Lets look at how the model does with using `balance`, `income`, and `student` as the predictor variables.

```
default.fit3 <- glm(def ~ balance + income + student, family = binomial, data = Default)

summary(default.fit3)
```

```
Call:
glm(formula = def ~ balance + income + student, family = binomial,
    data = Default)

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -10.869045   0.492256 -22.080  < 2e-16 ***
balance       5.736505   0.231895  24.738  < 2e-16 ***
income        0.003033   0.008203   0.370  0.71152
studentYes   -0.646776   0.236253  -2.738  0.00619 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2920.6  on 9999  degrees of freedom
Residual deviance: 1571.5  on 9996  degrees of freedom
AIC: 1579.5

Number of Fisher Scoring iterations: 8
```

```
predict(default.fit3,
        newdata = data.frame(balance = c(1,2),
                             student = c("Yes",
                                         "No"),
                             income=40),
        type = "response")
```

```
          1           2
0.003477432 0.673773774
```

```r
predict(default.fit3,
        newdata = data.frame(balance = c(1),
                             student = c("Yes",
                                         "No"),
                             income=c(10, 50)),
        type = "response")
```

```
          1           2
0.003175906 0.006821253
```

18 Logistic Regression Diagnostics and Model Selection

Things are a bit different in GLMs. There is no assumption of normality and the definition of a residual is much more ambiguous.

- We verify the linearity assumption, i.e., $g(\mu_{y|x})$ is linearly related to the predictors. (NOT $\mu_{y|x}$).
  - This one is a bit difficult to tease out.
- We check multicollinearty/VIF like before.
- Assess for influential observations.

18.1 Data: Do you have mesothelioma? If so call $< ATTORNEY >$ at $< PHONENUMBER >$ now.

18.1.1 Data description

From UCI Machine Learning Repository: https://archive.ics.uci.edu/ml/datasets/Mesothelioma%C3%A2%E2%82%AC%E2%84%A2s+disease+data+set+

This data was prepared by; >Abdullah Cetin Tanrikulu from Dicle University, Faculty of Medicine, Department of Chest Diseases, 21100 Diyarbakir, Turkey e-mail:cetintanrikulu '@' hotmail.com Orhan Er from Bozok University, Faculty of Engineering, Department of Electrical and Electronics Eng., 66200 Yozgat, Turkey e-mail:orhan.er@bozok.edu.tr

In order to perform the research reported, the patient's hospital reports from Dicle University, Faculty of Medicine's were used in this work. One of the special characteristics of this diagnosis study is to use the real dataset taking from patient reports from this hospital. Three hundred and twenty-four MM patient data were diagnosed and treated. These data were investigated retrospectively and analysed files.

In the dataset, all samples have 34 features because it is more effective than other factors subsets by doctor's guidance. These features are; age, gender, city, asbestos exposure, type of MM, duration of asbestos exposure, diagnosis method, keep side, cytology, duration of symptoms, dyspnoea, ache on chest, weakness, habit of cigarette, performance status, White

Blood cell count (WBC), hemoglobin (HGB), platelet count (PLT), sedimentation, blood lactic dehydrogenise (LDH), Alkaline phosphatise (ALP), total protein, albumin, glucose, pleural lactic dehydrogenise, pleural protein, pleural albumin, pleural glucose, dead or not, pleural effusion, pleural thickness on tomography, pleural level of acidity (pH), C-reactive protein (CRP), class of diagnosis. Diagnostic tests of each patient were recorded.

18.2 Model Selection

There are a lot of variables in the data...AND I don't have a code book for what some of the stuff means.

What do I do?

```
cols(
  age = col_double(),
  gender = col_double(),
  city = col_double(),
  asbExposure = col_double(),
  asbDuration = col_double(),
  dxMethod = col_double(),
  cytology = col_double(),
  symDuration = col_double(),
  dyspnoea = col_double(),
  chestAche = col_double(),
  weakness = col_double(),
  smoke = col_double(),
  perfStatus = col_double(),
  WB = col_double(),
  WBC = col_double(),
  HGB = col_double(),
  PLT = col_double(),
  sedimentation = col_double(),
  LDH = col_double(),
  ALP = col_double(),
  protein = col_double(),
  albumin = col_double(),
  glucose = col_double(),
  PLD = col_double(),
  PP = col_double(),
  PA = col_double(),
  PG = col_double(),
  dead = col_double(),
  PE = col_double(),
  PTH = col_double(),
  PLA = col_double(),
  CRP = col_double(),
  dx = col_double()
)
```

18.2.1 Step one: Assess your goal and how you get there.

The goal will be to predict the probability of the subject having mesothelioma:

- `dx` is 1 for mesothelioma, 0 if "healthy"
- I want to do this for people that are ALIVE. And I want to do it based on the characteristics of the person
  - The `dead` variable is therefore useless to me, I can get rid of it.
  - `dxMethod` doesn't seem useful for my goal by its name either.
  - I'm suspicious of `cytology` since it may be whether a specific type of test is used. That would not be helpful in actually predicting if someone has mesothelioma based on the attributes of that person.
- There are many categorical variables that I don't know how they are coded.
  - `city`, `smoke`, `perfStatus`
  - If I can't assess how they contribute to the model, I'm not going to use them.
  - I don't know how to measure those variables for future use of the model.
  - Also, as a statistician, I can't give meaningful results if I don't have information about a variable.

That still leaves of with 24 variables…

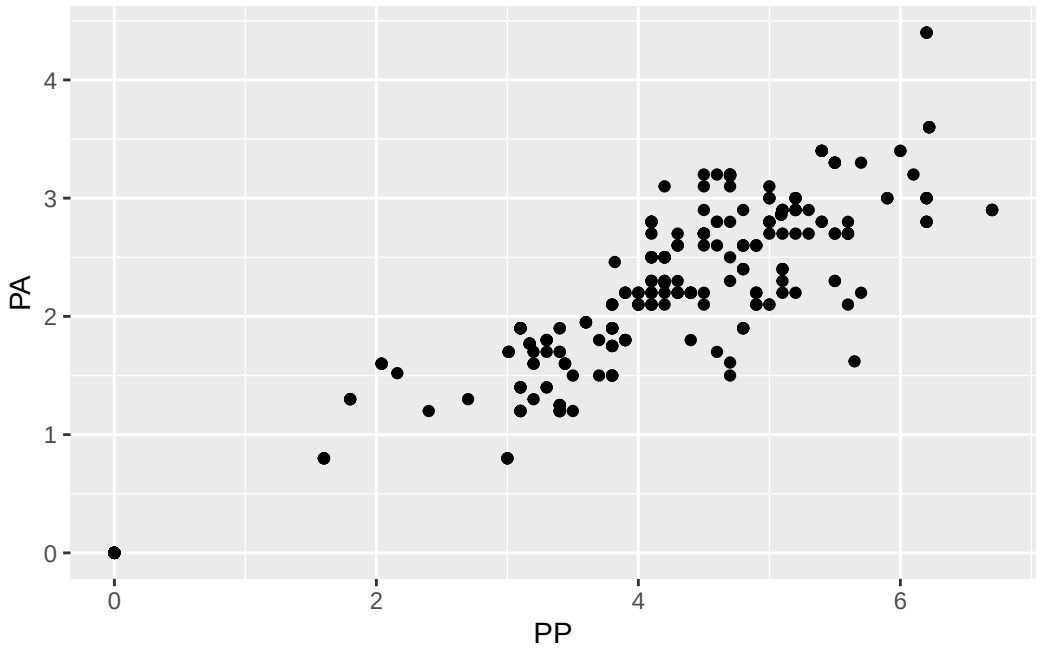
18.2.2 Variance Inflaction Factors are still here

Let's start with the full model.

Now get the VIFs (courtesy of 'car library)

```
      age        gender   asbExposure   asbDuration   symDuration
 1.623728      1.097621      2.719456      3.482200      1.141027
dyspnoea     chestAche      weakness            WB           WBC
 1.094622      1.062902      1.220221      1.119346      1.124898
      HGB           PLT sedimentation           LDH           ALP
 1.157734      1.886302      1.135334      1.870804      1.247170
  protein       albumin       glucose           PLD            PP
 1.336606      1.413671      1.102442      1.250738      7.939269
       PA            PG            PE           PTH           PLA
 7.117496      2.344206      2.351430      1.705495      1.878006
      CRP
 1.565218
```

- The criteria is about the same as before:
- If there are any VIFs greater than 5, investigate.
- If there are more than a few (no hard rule to give) in the 2 to 5 range or things seem weird, investigate.
- What strikes me is `absExposure` and `absDuration`.
  - These are whether someone has been exposed, and if so, how long were they exposed in days.
  - They are NOT independent of each other.
  - There are things to consider which one is wisest to remove.
  - `asbDuration` contains information about both, so maybe that's best.
- That leaves us mainly with `PP` (pleural protein?) and `PA` (pleural albumin?)
  - They seem to be fairly correlated with each other.
  - I have no idea what is more important (maybe you would!) so I'm going to take `PP` out.
  - A less arbitrary option may be to try two models, one with each missing, and see how well they fit/perform.


PA
PP
0
1
2
3
4
0
2
4
6

18.2.3 Check after removing variables

So lets cut the variables down a bit more and check how things are doing.

```
Call:
glm(formula = dx ~ ., family = binomial, data = meso3)

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)    1.227e+00  1.944e+00   0.631  0.52779
age           -4.656e-02  1.472e-02  -3.163  0.00156 **
gender        -7.519e-01  2.735e-01  -2.749  0.00597 **
asbDuration    2.945e-02  1.059e-02   2.781  0.00541 **
symDuration    5.788e-02  2.902e-02   1.994  0.04611 *
dyspnoea      -2.021e-02  3.484e-01  -0.058  0.95375
chestAche     -2.540e-01  2.830e-01  -0.898  0.36939
weakness       3.429e-01  2.964e-01   1.157  0.24726
WB            -1.577e-05  3.982e-05  -0.396  0.69201
WBC           -6.393e-02  4.185e-02  -1.528  0.12655
HGB            3.258e-01  2.844e-01   1.146  0.25195
PLT           -2.201e-03  1.140e-03  -1.931  0.05349 .
sedimentation  4.768e-03  6.457e-03   0.738  0.46024
LDH            4.156e-04  9.604e-04   0.433  0.66522
ALP           -8.684e-04  4.290e-03  -0.202  0.83961
protein        7.299e-02  1.800e-01   0.405  0.68512
albumin        6.421e-02  2.442e-01   0.263  0.79262
glucose        2.416e-03  3.496e-03   0.691  0.48940
PLD           -1.459e-04  3.582e-04  -0.407  0.68367
PA            -2.819e-01  1.862e-01  -1.514  0.12996
PG            -8.418e-03  7.333e-03  -1.148  0.25097
PE             1.416e-01  5.414e-01   0.262  0.79366
PTH            2.249e-01  3.475e-01   0.647  0.51747
PLA           -7.894e-01  3.567e-01  -2.213  0.02689 *
CRP            1.226e-02  7.395e-03   1.658  0.09730 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 393.79  on 323  degrees of freedom
Residual deviance: 348.20  on 299  degrees of freedom
```

```
AIC: 398.2

Number of Fisher Scoring iterations: 5
```

age	gender	asbDuration	symDuration	dyspnoea
1.434130	1.088232	1.487264	1.129502	1.094376
chestAche	weakness	WB	WBC	HGB
1.045581	1.178690	1.115586	1.096017	1.157860
PLT	sedimentation	LDH	ALP	protein
1.858685	1.140208	1.871628	1.230647	1.306394
albumin	glucose	PLD	PA	PG
1.344529	1.098984	1.235294	1.800686	2.337586
PE	PTH	PLA	CRP	
2.122909	1.701094	1.857195	1.558289	

Everything is way better in terms of VIF so that's a good first step I'd say.

18.3 Residual Diagnostics

18.3.1 Review

Assessing the residuals is a bit difficult here, originally it was pretty simple.

$$e_i = \text{observed} - \text{predicted} = y_i - \hat{y}_i$$

And then there are standardized residuals and studentized residuals (which get called standardized residuals because someone was sadistic).

$$e_i^* = \frac{y_i - \hat{y}_i}{\sqrt{MSE}} \qquad \text{or} \qquad e_i^* = \frac{y_i - \hat{y}_i}{\sqrt{MSE(1-h_i)}}$$

- Both have the same objective: set the residuals on the same scale no matter what model you're examining.
- Standardized/Studentized Residuals greater than 3 are considered outl;iers and the corresponding data point should be investigated.
- Studentized residuals (the right hand-side version) are calculated in such a way that they try to account for over-fitting of the model to the data by pretending that observation is not in the data.
- It is my impression (and opinion) that Studentized should be preferred.

18.3.2 Residuals in GLMs

Residuals follow the same form.

$$e_i = \text{observed} - \text{predicted} = y_i - \widehat{y}_i$$

The general form for a standardized residual is about the same:

$$e_i = \frac{y_i - \widehat{y}_i}{s(\widehat{y}_i)}$$

and $s(\widehat{y}_i)$ is the variability of our estimate.

- In GLMs it is often the case, $s(\widehat{y}_i)$ is not constant by default, i.e., the variability of the predictions is not constant.
- Dependent on our model, the way to calulate $s(\widehat{y}_i)$ differs.

18.3.3 Pearson Residuals

We have to account for the fact that our predictions are probabiliies in logistic regression.

$$\hat{y}_i = \widehat{P}(y_i = 1|x) = \hat{p}_i(x_i)$$

We are dealing with the binomial distribution so the formula for the variability of the predicted probability is:

$$s(\hat{y}_i) = \sqrt{\hat{p}_i(x_i)(1 - \hat{p}_i(x_i))}$$

Therefore the Pearson standardized residual is:

$$e_i = \frac{y_i - \hat{p}_i(x_i)}{\sqrt{\hat{p}_i(x_i)(1 - \hat{p}_i(x_i))}}$$

If you can group your predictions somehow in groups of size n_i, then the pearson residual is

$$e_i = \frac{y_i - \hat{p}_i(x_i)}{\sqrt{n_i\hat{p}_i(x_i)(1 - \hat{p}_i(x_i))}}$$

18.3.4 Deviance residuals

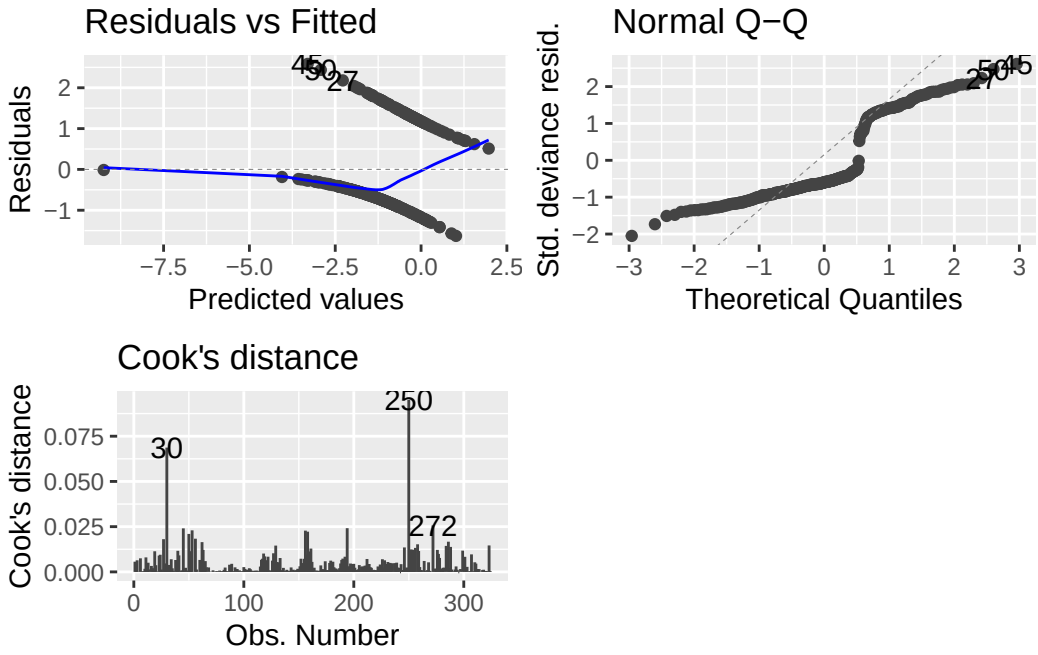
There is something called a deviance residual, for which the formula is:

$$e_i^* = sign(y_i - \widehat{y}_i)\sqrt{2[y_i \log(\frac{y_i}{\widehat{y}_i}) + (n_i - y_i)\log(\frac{n_i - y_i}{n_i - \widehat{y}_i})]}$$

- The formula is not very illuminating unless you have deeper theoretical knowledge.
- I have seen at least one person say this is "preferred".
- I have also seen that examining them is about the same as examining the pearson residuals.
- An advantage here is that deviance residuals are universal and relate to some of the deeper structure of GLMs.

18.3.5 Autoplot maybe?

We can examine the residuals via autoplot and assess outliers as well..



- Technically, these plots indicate that nothing is wrong.
  - There is a lot of intuition and theory type stuff for why I can say that.
  - That might be a whole more week's worth of material, but we are cutting here.
  - Consider it a self study topic.
  - Hopefully, I've given enough of a seed of information for you to teach yourself more in depth modeling.
  - If you want to be good, the learning never ends.

Anyway, on to the plots.

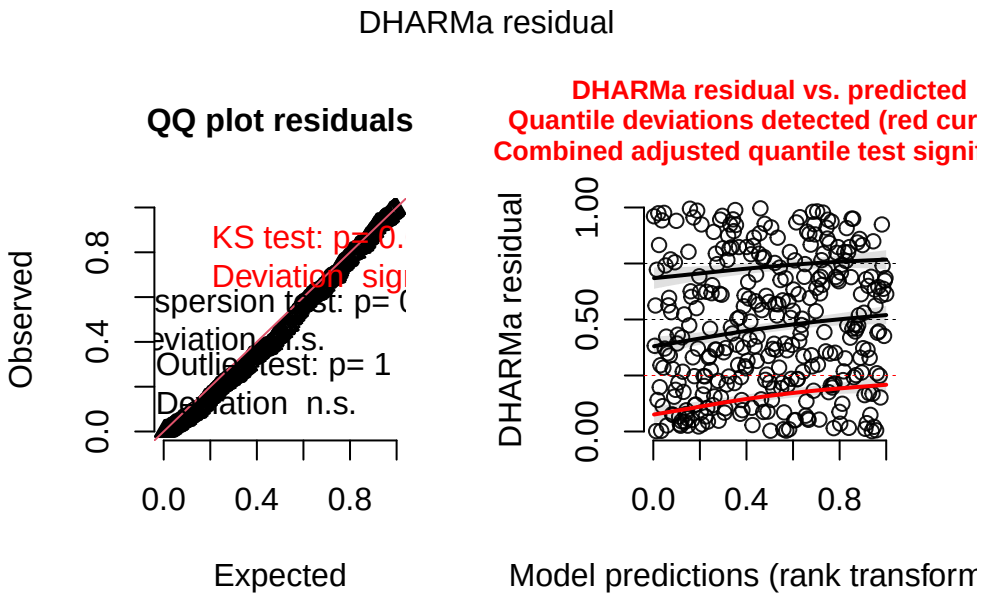
- Top-left may not be familiar since our observed values are either zero or one.
- QQ plot looks weird because there is not a normal distribution to expect, necessarily.
- Cook's distance is at least familiar.
  - 0.5 denotes a potentially problematic value.
  - 1 is the cutoff for an extreme outlier.
  - Some people say 4/n is a good cutoff... I have found no reason to back that up so far.
- Honestly I am still at a loss for how useful examination of the plots can be. GLMs can really mess with traditional methods.

18.3.6 DHARMa package for plotting residuals.

An experimental way of examining the residuals is availbale via the `DHARMa` package.

> Florian Hartig (2021). DHARMa: Residual Diagnostics for Hierarchical (Multi-Level / Mixed) Regression Models. R package version 0.4.1. https://CRAN.R-project.org/package=DHARMa

- It tries to streamline the process and make it easier for non-experts to examine residuals via simulation.
- There are only a couple of functions you need to know.
  - `simulateResiduals(model)` which is stored as a variable.
  - `plot(simRes)` where `simRes` is the simulated residuals output variable.
- Because it runs simulations (repeated calculations), it may take a while to finish.



Notice the output basically tells you what may or may not be wrong. It's as simple as that.

- The QQ Plot is NOT for normality.
  - BUT you still want to check if they follow a straight line.
  - The KS Test p-value is for the test of whether the data are following the correct distribution or not.
  - Small p-values mean there are problems with your model.

- Here everything seems ideal.
  - The QQ-Plot is a nearly straight line.
  - The residual plot on the right follows the pattern we'd want. Everything is centered at 0 and the spread is consistent.
- I cannot 100% recommend that this is all that you rely on, I am merely proposing a tool to get an overview of the situation.
- In reality there are a LOT of nuances to what's going on, so be wary.
- As far as the relative simplicity of the models we look at, my guess is that this is fine to use.

18.4 Marginal Effects

If you haven't picked up on it yet, interpreting the logit link model is rather hard in comparison to a "normal" linear regression model.

On top of reporting the model coefficients, we should probably report the marginal effects as well.

The marginal effects are, in their most basic forms, calculations derived from the model not represented by the coefficients alone.

They take many forms.

18.4.1 What are marginal effects

Suggested Read: https://www.andrewheiss.com/blog/2022/05/20/marginalia/

> Statistics is all about lines, and lines have slopes, or derivatives. These slopes represent the marginal changes in an outcome. As you move an independent/explanatory variabl, what happens to the dependent/outcome variable?

Think back to the last unit where we calculated the "marginal means" from the ANOVA using the `emmeans` R package. These were important because sometimes they are *not* equal to the summary statistics.

- Marginal effect: the statistical effect for continuous explanatory variables; the partial derivative of a variable in a regression model; the effect of a single slider
- Conditional effect or group contrast: the statistical effect for categorical explanatory variables; the difference in means when a condition is on vs. when it is off; the effect of a single switch
- Average marginal effect vs Marginal effect at the Mean

Characteristic	OR[1]	95% CI[1]	p-value
tx			
tPA	—	—	
SK	1.24	1.12, 1.37	<0.001
age	1.07	1.07, 1.08	<0.001
Killip Class			
I	—	—	
II	2.10	1.88, 2.35	<0.001
III	4.70	3.73, 5.89	<0.001
IV	20.8	15.6, 27.8	<0.001
Previous MI			
no	—	—	
yes	1.71	1.54, 1.91	<0.001
MI Location			
Inferior	—	—	
Other	1.36	1.04, 1.75	0.020
Anterior	1.78	1.62, 1.96	<0.001
Sex			
male	—	—	
female	1.41	1.27, 1.56	<0.001

[1]OR = Odds Ratio, CI = Confidence Interval

- It matters a lot outside of normal linear regression

18.4.2 Example Data (real study)

So why do you need to know about this: interpretation.

Let's use an example from the GUSTO study.

Let's load the packages and study data.

So we have an effect… but what does an odds ratio really mean? Is it big or small?

Using the marginal means, we can calculate the risk difference instead which is adjusted for the covariates. The treatment effect *on this scale* will vary based on the covariate values.

We can then calculate the average treatment effect on the probability (response scale) with the following:

```
 Estimate Std. Error    z Pr(>|z|)    S   2.5 % 97.5 %
   0.0119    0.00281 4.24   <0.001 15.5 0.00641 0.0174

Term: tx
Type:  response
Comparison: mean(SK) - mean(tPA)
Columns: term, contrast, estimate, std.error, statistic, p.value, s.value, conf.low, conf.hig
```

This study, with this model, shows roughly at 1.11% reduction in risk.

We could also talk about risk in terms of relative effect. These are called risk ratios:

```
 Estimate Pr(>|z|)    S 2.5 % 97.5 %
     1.19   <0.001 14.7   1.1    1.3

Term: tx
Type:  response
Comparison: ln(mean(SK) / mean(tPA))
Columns: term, contrast, estimate, p.value, s.value, conf.low, conf.high, predicted_lo, predi
```

Shows about a 1.18 times higher risk in the SK treatment group.

18.4.3 Average Effect at the Mean

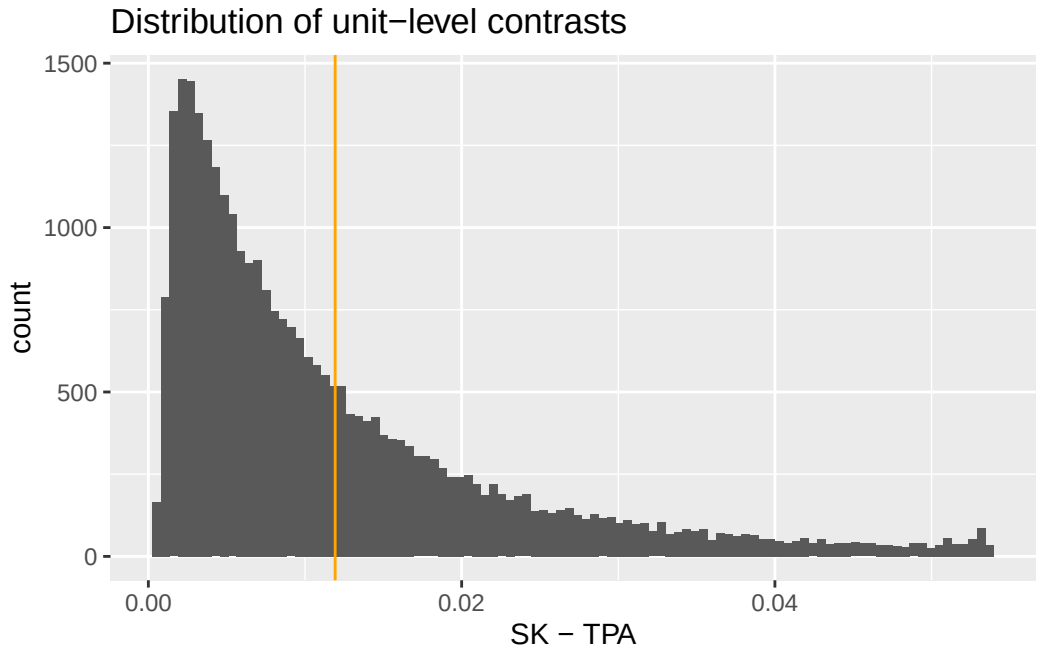
This is what you get in most software.

```
 Estimate Std. Error    z Pr(>|z|)    S   2.5 % 97.5 %  age Killip pmi    miloc
  0.00506     0.0012 4.22   <0.001 15.4 0.00271 0.0074 60.9      I  no Inferior
  sex
 male

Term: tx
Type:  response
Comparison: SK - tPA
Columns: rowid, term, contrast, estimate, std.error, statistic, p.value, s.value, conf.low, c
```

UH OH! The effect is *substantially* very small compared to the average effect. Why? This is for a data point that doesn't exist!

18.4.4 Individual level summary

Again, the effect varies on individual level based on the effect of other covariates. This is more extreme when the base probability is very low (rare event).



18.4.5 What about continuous variables?

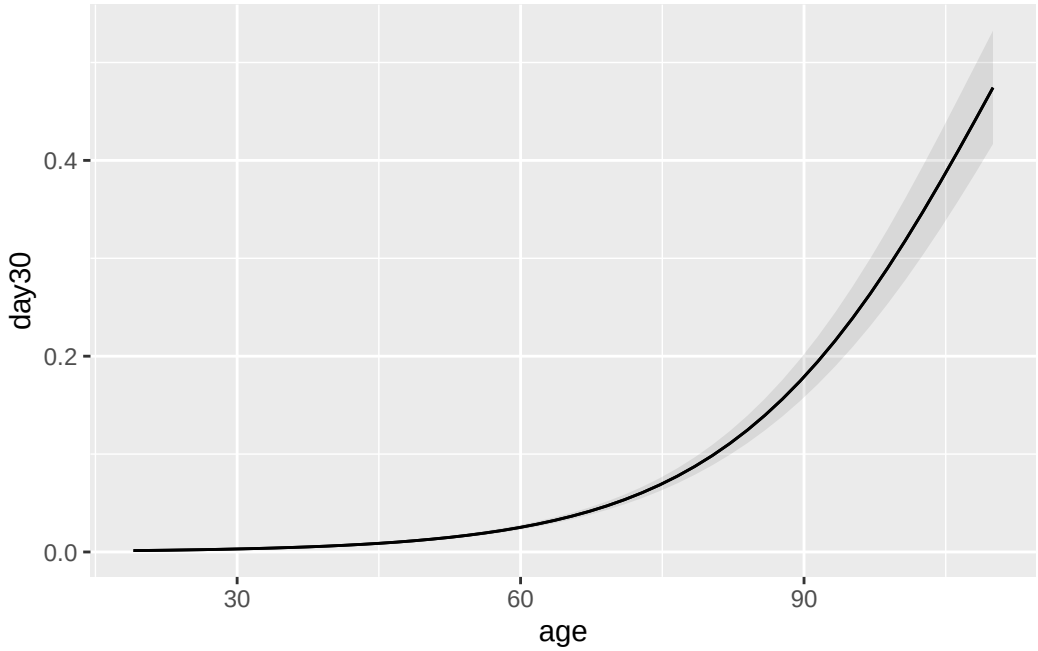
We can look at the slopes.

Let's look at age!

```
 Estimate Std. Error    z Pr(>|z|)     S   2.5 %  97.5 %
  0.00406   0.000149 27.3   <0.001 543.6 0.00377 0.00435

Term: age
Type:  response
Comparison: mean(dY/dX)
Columns: term, contrast, estimate, std.error, statistic, p.value, s.value, conf.low, conf.hig
```

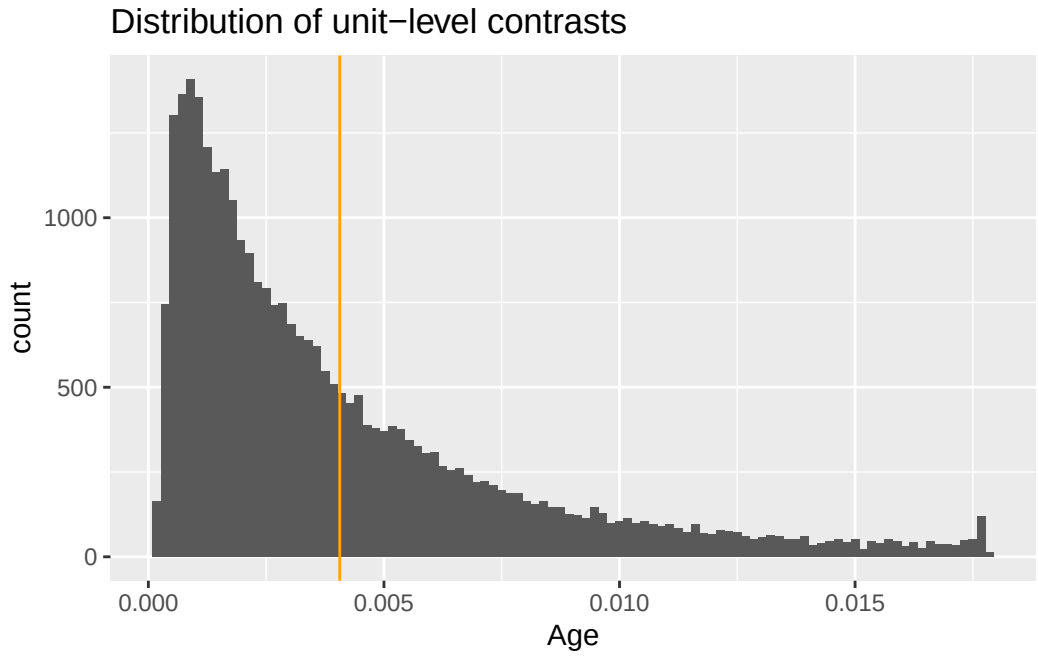
So approximately at .4% increase in risk per year increase in age

Let's plot this to see what it looks like.



That doesn't look linear…. Why? We model the *log odds* not the probability directly.

We can look at all the individual level slopes as well.



References

Balka, Jeremy. n.d. "Making Statistics Make Sense." *JB Statistics*. https://www.jbstatistics.com/.